Ghent University
Faculty of Sciences
Department of Applied Mathematics, Computer
Science and Statistics

Explicit sequence-culture-strain-taxon links in StrainInfo and their role in quality assessment and assurance

Wim De Smet

Supervisors: Prof. Dr. Peter Dawyndt
Prof. Dr. Bernard De Baets
Prof. Dr. Paul De Vos

Dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Computer Science.

Voor opa en pepé.

Dankwoord

Zonder de steun van collega's, vrienden en familie zou dit doctoraat niet mogelijk geweest zijn. Bij deze wil ik dan ook iedereen bedanken die mij over de jaren geholpen of bijgestaan hebben. Het is onmogelijk om hier iedereen op te lijsten, maar weet dat ik elke hulp, hoe klein ook, zeer geapprecieerd heb.

In het bijzonder zou ik graag mijn promotoren Peter, Bernard, en Paul bedanken omdat ze hun vertrouwen in mij gesteld hebben voor het volbrengen van dit werk, en voor hun hulp bij de vele aspecten ervan. Samen met hen en de rest van het StrainInfo team hebben we StrainInfo gemaakt tot wat het vandaag is. Nu, 4–5 jaar later, denk ik soms nog altijd nostalgisch terug aan de vele technische discussies en late vergaderingen samen met Bert Verslyppe en Wim Gillis (en, geef toe, het vele gepruts) die leidden tot de eerste rewrite van StrainInfo. Bedankt ook, aan de sysadmins die alles draaiende na het vertrek van Wim, met name Kenneth Waegeman, Bert Sarens en Bart Verheyde.

Tijdens dit doctoraat was ik als informaticus ingebed in het labo microbiologie. Allereerst dank aan Margo, die altijd klaarstond om mij wegwijs te maken, en die je altijd kan aanspreken voor problemen. Extra dank ook aan de mensen op mijn gang, in het bijzonder Bart Hoste met wie ik over veel gepraat heb, soms ook over biologie, en Geert Huys voor zijn hulp bij mijn verwerking van de biologische achtergrond. Voor de vele gesprekken die mijn tijd aan LMG leven gaven ook bedankt aan Jonas en Ines, die mij nog begeleid hebben in mijn eerste vak microbiologie, Lies en Katrien, voor de epic namiddagpauzes nu al een aantal jaar geleden, en Sven, Joanna, Gwen en Sofie, met wie ik vaak de Tobbe onveilig maakte, en Jindrich voor de looptraining. Extra dank aan Joke, zonder wie ik nooit zou leren dansen hebben en op wiens tenen ik bijna nooit gestapt heb. Ook de vele collega's in TWISt en KERMIT mag ik hier niet vergeten. Extra dank aan Chris en Patricia voor hun begeleiding bij mijn laatste master thesis, en Karel en Bart met wie het een plezier was samen les te geven.

A big thank you as well to my colleagues who don't read Dutch, especially to the people of the GSC, who always made me feel welcome, and in particular

Frank Oliver Glöckner and Renzo Kottmann, with whom I enjoyed working together and who welcomed me with open arms to Bremen. A special thank you for Mario Amalfi, who took me in and showed me exactly how a mycologist works, still one of my most positive experiences within our work with the IUAP.

Collega's kunnen je wel helpen tijdens het werk, maar je vrienden houden je gaande daartussenin. Het ging niet altijd zo vlot met dat programmeren en schrijven, en zij zullen het geweten hebben. In het bijzonder wil ik Willem, Jonathan, Jannick en Lieven bedanken voor hun steun en luisterbereidheid. Ook de rest van de kliek van Zottegem wil ik bedanken voor hun geduld tijdens mijn afwezigheid dit laatste jaar, met name Sebastiaan, Joris, Yven, Jenda, Iris, Steven, Filip en Femke.

For those friends who don't speak Dutch, thank you from the bottom of my heart. Jon, thank you for always listening to me when I had something to say, and for introducing me to English beer and cooking (in that order). Thanks to Tea and Hannu, for welcoming me in their home and introducing me to Finnish beer and cooking (I sense a pattern) and Mark and Sarah for lots of fun nights, even as we slightly trashed your home. Special thanks to Mark Macciocchi, for a lot of late night conversations in those days when I was still mightily addicted to MMO's. For that matter, thanks to all the members of Overbuffed & Underpaid, you guys rock! Miro and Denitsa, you were always up for a party, even when it was a bad idea, but I sure enjoyed it. Thanks for being my friends. Finally, big thank you to Mesude for some encouragement when I needed it most.

De belangrijksten komen altijd laatst, en dat is hier ook niet anders. Dank aan mijn broer, Tom, en zussen, Cindy en Ilse, om er altijd voor mij te zijn wanneer ik jullie nodig heb. Dank aan mijn ouders, zonder wie — een cliché, ik weet het — ik hier niet zou zijn. Niet alleen omdat ik hun zoon ben, maar vooral omdat ze mij mijn hele leven lang al graag zien en op elke mogelijke manier gesteund hebben. Het was thuis dat ik spelenderwijs mijn eerste stapjes zette in de informatica, en met hun hulp leerde ik ook de schoolse kant ervan appreciëren. Maar het belangrijkste dat ik van thuis meegekregen heb is dat geborgen gevoel dat je alleen van liefhebbende ouders kan krijgen. Mama, papa, ik hou van jullie.

Spijtig genoeg heeft niet iedereen het geluk dit doctoraat nog mee te maken. Begin dit jaar kwamen mijn beide grootvaders te overlijden. Hoewel ze een lang en gelukkig leven geleid hebben, voelde het nog veel te vroeg. Daarom, en in de wetenschap dat hun harde werk twee generaties later nog resoneert, draag ik graag dit werk aan hen op.

Last, but definitely not least, I would like to thank my fiancée, Leilei, for loving and supporting me during these past few years. It was often not an easy time for me, and I know this must have made things hard on her as well. Still, she was there when I needed her most, always ready to give advice and help me keep my head above water. Without her, this book would have never been finished. For that and much more, I will always be thankful to her.

李蕾蕾，我爱你。

Gent, 15 november, 2013
Wim De Smet

Contents

1 Introduction

1.1 Short introduction to microbiology

The first cellular life on our planet started small, with micro-organisms that have sizes on the order of a few micrometers ($1\mu m = 10^{-6}m$ or $1/10,000th$ centimeter). These organisms were the first to appear, evolve and (though much later) the first to start oxygenating our atmosphere. After larger single-celled organisms and multicellular life appeared, microbial life did not vanish. It remained growing and evolving, in extremely diverse habitats all over the planet, residing in the soil, the water, the air, on or inside rocks, plants and animals.

Even though they are invisible to the naked eye, microbial organisms of every shape and form can be found anywhere life is possible. As a result, they greatly outnumber any other organism on earth. They contain an equally large amount of the world's biomass, close to matching the total organic carbon stored in plants [1] and greatly exceeding that stored in other forms of life. It is no surprise then, that micro-organisms play a large role in many of the world's ecosystem and impact our lives in myriad ways, even if we are not always aware of it. To the general public, microbes of all kinds are perhaps best known as the live agents behind many infections and diseases. However, their main

role in our life is beneficial. Closest to home are the beneficial microbiota of our gut, without which we would not be able to break down our food, and the microbiota of our skin, that help keep it moisturized and defend against colonization by other species. Further afield are the microbes in agriculture, that play a central role in the fixation of nitrogen, or in food production where fermentation processes are widely used, and in their use in various industrial processes [2].

In biodiversity/taxonomy, living organisms were traditionally divided into two main groups: the prokaryotes, comprising single-celled organisms which have their genetic material floating free inside a membrane and possibly a cell wall, and the eukaryotes, in which the genetic material is encapsulated in an inner nucleus. Evolutionary, however, it has become clear over time that life can be subdivided in three different domains [3]: the Bacteria, Archaea and Eukarya. While members of the domains of Bacteria and Archaea are prokaryotes (i.e. their genetic material is free-floating inside the cell), evolutionary the Archaea actually appear to be closer to eukaryotes than to Bacteria [4, 5]. As such, the term prokaryotes is deemed by some to be a misnomer, but it retains value as a collective term for the two domains of Bacteria and Archaea, and is used in this thesis as such. Much of the focus of this work is on prokaryotes, although StrainInfo also provides support for cultures and taxonomy of fungi (which are eukaryotes), see Chapter 2.

1.1.1 Establishment of modern microbiology

Various theories of disease causing agents predate their discovery, but even though microbes are quite literally omnipresent on this planet, it would take until the 17th century before the invention of the microscope would allow scientists to study these organisms in earnest. Invented by a Dutch draper by the name of Antonie van Leeuwenhoek (Figure 1.1), it was the first real tool of the field of microbiology. Even then, it took until the 19th century before techniques were developed to reliably grow micro-organisms in the lab and isolate them from each other, leading to so-called pure cultures of micro-organisms. Such a pure culture allows for extensive study of bacteria in isolation. Much of modern microbiology still relies on these techniques for isolation of new micro-organisms and study of existing known diversity.

From the very beginning, researchers had been classifying bacteria, assigning names and proposing theories for their evolutionary relationships, establishing

Figure 1.1: Portrait of Antonie van Leeuwenhoek, the first person to observe microorganisms by microscope. Portrait by Jan Verkolje, 1686, Object number RP-P-OB-17.513, courtesy of Rijksmuseum (Netherlands) under educational licence.

the fields of microbial taxonomy and bacteriology. With this practice came the identification techniques developed over the years, many of which are still actively used in (medical) microbiology today. Originally, researchers could only rely on what could be observed directly from the cells, from the general shape and colour of a microbial colony (a collection of millions of microbial cells), to the vague internal structures that were visible under a (light) microscope. Later imaging technology would greatly improve on this ability, but it was the addition of (bio)chemical and metabolic tests eventually that lead to sophisticated assays for the identification of a range of micro-organisms. Even so, the resolution of all these techniques is limited, and many new avenues were explored over the years for more accurate identification and deduplication techniques.

In order to safeguard the work developed over the years by researchers in this field, they devised ways to store the source material, namely the cultures they grew and studied in the lab. Over time, specialized institutions appeared, here referred to as Biological Resource Centers (BRCs), to safeguard and store large quantities of such biological material. Most resource centers are service providers, providing organisms to researchers for study or for industrial applications. In this role they are the gatekeepers to much of the known bacterial diversity for the future, allowing researchers to repeat previous studies and come up with new studies or applications.

1.1.2 Molecular biology

In the twentieth century would come the discovery of the true nature of the genetic code that determines most of the phenotype of all organisms, as stored in one or more deoxyribonucleic acid (DNA) molecules. The whole of genetic information contained in a cell is referred to as the genome of that organism. A DNA molecule takes the form of a double-stranded helix, of which each strand is a chain of nucleotides, the informational molecules that make up the genetic code. The information content is actually determined by the type of nucleobase (one of adenine, thymine, cytosine or guanine), being one building block of a nucleotide, and as such individual residues in a strand are often referred to as bases or base pairs, as they always occur paired with a complementary base on the opposite strand within the DNA helix.

Individual parts of the genome from one single functional unit form genes. While micro-organisms do exchange genes in other ways than through cell

division (in so-called Horizontal Gene Transfer between possibly unrelated cells), much of the evolution of the genes fundamental to the operation of the organism or the metabolic processes it performs happens vertically, i.e. from mother to daughter cells. Many of these genes are so fundamental to the operation of a cell that they occur in all individuals of their species and often whole lineages. What's more, theories of evolution posit that genes will undergo mutation: a gradual change of parts of the genetic code for that gene over time. Comparing the two versions of a gene in different organisms thus can be used as a way to determine the evolutionary distance between both organisms, provided the gene was passed through vertical gene transfer. An example of one of these genes, the rRNA gene coding for the 16S portion of the small subunit of the ribosome, will be discussed in Chapter 5. The use of DNA relatedness and phylogenetic markers has been instrumental to the progress of the fields of taxonomy and systematics through phylogenetic study (the study of evolutionary lineage through use of genetic methods). They also provide for useful identification methods, though other lower cost methods remain popular.

Over time, the technology to extract and read the sequence of genes from microbial organisms has greatly improved and spread widely. As it quickly became standard to extract certain genes from organisms, for instance in phylogenetic studies, a need soon developed for places to store these sequences, since their length and number became thus that they could simply not be reported directly inside publications anymore. Therefore, several databases were set up to hold this information. The three biggest ones eventually started collaborating, creating the International Nucleotide Sequence Database Collaboration [6], which freely stores and distributes sequences to all interested parties.

1.2 Connecting physical and virtual databases

Each BRC documents the total of its holdings in a catalogue: a list of organisms published on paper or online, that lists for each organism its accession number, properties and taxonomic name, and provides a way for users to purchase it. These catalogues are by their nature distributed, as each is maintained by an independent BRC and there is no central source of information on the total availability of strains. Likewise, publications end up on the websites of a number of relevant journals, often referred by title or by an identifier in

bibliographical databases such as PubMed[1]. The molecular sequence information reported in papers is stored in the three databases of the International Nucleotide Sequence Databases Collaboration (INSDC) under a specific accession number. In addition to all this, databases have been established for the cataloguing of certain important aspects of the publication record, such as the taxonomy databases that record the current status of bacterial and fungal taxonomy.

All different databases involved are isolated from each other, occasionally referring to objects in other databases, but seldom linking the two data records together explicitly. This is the innovation StrainInfo brings to this problem: by piecing together the information from different BRC catalogues, it is possible to build overview lists of all subcultures of a particular microbial strain stored in BRCs world-wide (Chapter 2). This makes it much easier for researchers to find subcultures of their strain of interest, even if all they know is the strain number of a culture of the same strain stored halfway around the world. These strain numbers are the identification numbers assigned by BRCs and researchers to specific cultures stored locally. They are often mentioned on sequence records from the molecular sequence databases, so by using the set of strain numbers for each strain, it also becomes possible to explicitly determine which sequence records have been derived from the same strain. Likewise, genome projects (sequencing projects that seek to sequence the entire genome of an organism) can be linked to a strain, and further information on the projects can then be linked to the strain using genomic databases that map their data to the project in the Genomic Rosetta Stone (Chapter 4). Chapter 3 describes how this process has been achieved in StrainInfo.

In this way, it has become possible to link a taxon with its type strain. A strain consists a set of cultures stored in BRCs, and these cultures are themselves referred to by sequence records. Exploiting this chain of links from taxon to sequence record makes it possible to ask new questions and construct new applications, one of which is documented in the last and most important part of this Ph.D.: to find sequences for a particular strain automatically (described in part in Chapter 5), even using simply the name of the relevant taxon, while bypassing the manual lookup work so often typical of microbiological practice, and to use that knowledge to enable the development of the SeqRank workflow, described in Chapter 6. This application has opened up new ways to perform

[1] `http://www.ncbi.nlm.nih.gov/pubmed/`

data collection tasks that today many researchers still perform manually, and ways in which data quality can be checked and errors detected, simply by explicitly linking several databases together. This work explores the background of these approaches, the challenges involved in making them work and how explicit data links and the SeqRank workflow can be used to check for data consistency and quality, as well as build large representative sequence databases with less effort.

2 StrainInfo

2.1 Introduction

Exchange between researchers of microbial strain subcultures has underpinned microbiological science from the beginning. It is a prerequisite for reproducing and extending research performed on microbial resources that the physical material can be exchanged among scientists under appropriate quality and conditions. As the field of biotechnology has grown, these same organisms have likewise become increasingly important for industry and agriculture. Microbial strains have always been stored in private collections belonging to individual researchers or labs. Often, especially in the case of bacterial and fungal specimens, such a collection may have held living specimens. In many cases, collection have also consisted of dried (typically freeze-dried) or frozen (using cryopreservation techniques) samples. On request, microbial strains can be rehydrated or defrosted, grown in the lab, and delivered to interested researchers.

Local storage and exchange of microorganisms is still practiced to this day. However, it poses serious issues for the availability and authenticity of the original biological materials. Firstly, since growing and distributing strain subcultures can be a full-time job, researchers may not have the time or inclination

to honour requests from other parties, jeopardizing reproducibility of research. Secondly, individual researchers and labs do not normally specialise in long-term preservation of microorganisms, which raises the concern of important strains perishing or being destroyed. Thus, while private culture collections continue to play a significant role, researchers often exchange strains through the medium of public culture collections. These collections, which have been around since at least the late 19th century [7], specialise in all tasks surrounding the long-term storage of microorganisms or related materials and provide the necessary continuity in expertise not generally found in private collections. They allow the accession of microbial material, from both researchers and corporate entities, under generally agreed upon legal principles. Upon accession, an identifier is assigned to the strain, noting the collection it was entered in (usually by its acronym), as well as a number identifying it within the collection. This may later be used to refer to the strain, for instance in publications, and used by others to order subcultures of the same strain.

Through their role as specialised and independent collections of biological material, public culture collections are pivotal in ensuring the continuity of scientific research, and the availability of source material for industrial or agricultural applications. Furthermore, these public collections are crucial for providing reference microbial material (e.g. type strains) to studies in microbial taxonomy. In the case of prokaryotic type strains, accession of a new species in at least two public culture collections in different countries is mandatory for publication, in order to avoid permanent loss of biological material after publication and ensure material availability for research and industrial purposes [8]. To distinguish professional public culture collections from informal research collections, they are referred to here as Biological Resource Centers (BRCs), an umbrella term suggested by the OECD [9], which signifies a standard of quality assurance and the professional expertise necessary for safe long-term storage and exchange of microbial resources.

Because of their role as public repositories, BRCs often build up a large collection of economically or scientifically important organisms. However, distributing subcultures of their holdings world-wide poses both legal and practical issues in terms of safe storage and handling of the specimens, and the legal issues surrounding transfer of live samples of dual-use or economically valuable specimens. Considering the difficulties involved, researchers would normally prefer to obtain cultures from collections that are geographically close. Likewise, BRCs stand to gain from being able to offer the widest possi-

ble selection of strains in their chosen niche. Therefore, BRCs often agree to exchange materials between each other, under material transfer agreements (MTAs), which govern the legal peculiarities involved. This practice further strengthens the microbial commons by allowing end-users more ready access to the organisms involved. Furthermore, it ensures more copies of the biological material are in rotation, such that a problem of contamination or death of a strain in one collection does not jeopardize the overall availability of the original material.

BRCs exchange strains through bilateral agreements, the afore-mentioned MTAs. As a result, there is no central record of the locations a strain might have been transferred to. Each BRC does assign its own identifier to the strain it has received, and records a (minimal) history of identifiers other BRCs have used to refer to it. Unfortunately this means that there may be many different subcultures of a strain, all known under different strain numbers, each strain number referring to equivalent biological material. Since each BRC only records the linear history leading up to the original strain accession in its holdings, the full history of a strain is usually not available to the BRC, or its clients. Likewise, publications of research on the original material usually only refer to the source material by the strain number issued by the BRC it was derived from. This record is static, and is not updated later when the strain is further reproduced and transferred to other BRCs. This poses a quandary for end-users, who will have no choice to either laboriously search their local BRC catalogs, one by one, for their desired strains (using what history is provided), or to order the strain from whatever BRC they can find in the publication record, possibly facing high cost and delays, especially in the case of international transfer.

Fortunately, the history of individual transfers is available publicly in BRC catalogs, and so it is possible by examining these history records to build a centralised list of synonymous strain numbers for any particular strain [10]. When all strain numbers are known, it becomes possible for end-users to determine availability in a BRC of their choice much more accurately. Furthermore, strain numbers are often linked to additional information of importance to researchers. For one, They are cited in molecular sequence records in the databases of the International Sequence Database Collaboration (INSDC) [6]. Since different strain numbers might have been used, it is extremely hard to find all sequence records relating to a particular strain without access to its full synonymous strain number list. When this list is available, a list of sequence data relating to a particular strain can easily be constructed. Likewise, publi-

cations are often listed as part of those same sequence records, or as part of the information entries in BRC catalogs. Integrating this data further reveals information about the entries. Taxonomic data, available from many sources, e.g. the List of Prokaryotic Names with Standing in Nomenclature (LPSN) [11] or the original sequence records, also often mentions strain data and can likewise be integrated with the greater whole. All of these pieces can be linked to each other individually by gathering data and ensuring standards of quality assurance, gathered in a fully ontegrated database. For end-users to effectively make use of such an integrated database, however, a platform must exist for them to navigate the data and extract necessary information. The StrainInfo platform was created to supply these essential services, and offers a truly integrated view of strain related information for cultured strains of Bacteria, Archaea and Fungi.

Personal contributions to the StrainInfo platform

The StrainInfo platform as described here has been the work of several contributors. The initial database design, data and what legacy functionality is still in use today, were developed by Prof. Dr. Peter Dawyndt. A previous web version was co-developed by him and Bart Van Brabant. The current StrainInfo platform started as a complete redesign and shares no code with this original website. It has since long subsumed the original functionality. The core StrainInfo platform, namely the strain, publication, sequence and taxon passports, and assorted browsers (described below), was jointly developed in the course of this work by Dr. Bert Verslyppe [12], Wim Gillis and myself. Maintenance has over the years been performed by Wim Gillis, Bert Sarens, Kenneth Waegeman and Bart Verheyde.

Since StrainInfo core functionality was jointly developed, none of its parts can be wholly attributed to any of the three original authors, and in most cases except as noted in the text below, all these components were a group effort, garnering input from all members of the team. However, aside from minor additions and changes by others, several components of StrainInfo functionality were developed as part of this work, namely:

- Genome project passports and browser, developed as an application of the Genomic Rosetta Stone project (Section 2.2.6 and Chapter 4);

- StrainInfo Web Services: design and functionality that has enabled querying of StrainInfo through Simple Object Access Protocol (SOAP) web services (Section 2.5.1);
- StrainInfo original URI design: concept and early implementation (Section 2.5.2);
- StrainInfo custom exports: functionality and automation for the provision of large data exports to StrainInfo users;
- Development of an automated pipeline to streamline database updates, and development of several data filters, namely for downloading and importing into the database up-to-date molecular sequence information from the European Nucleotide Archive (ENA), and for the downloading and parsing of taxonomic information from the List Of Prokaryotic Names with Standing in Nomenclature (described in more detail in Chapter 3).

The work on StrainInfo and GRS described in the first chapters provides the explicit links that form a necessary basis for the building of further applications. A substantial portion of this work consists of the description of such an application for the ranking of 16S rRNA gene sequences, called SeqRank, and the Seqrank extended workflow. The background to the ranking algorithm used and the design of the parameters is described in Chapter 5, while the design of the application itself is described in Chapter 6. These latter two chapters form the main contribution of this work to the state of the art.

2.2 Passports and Browsers: Navigating strain related information

2.2.1 Passports as integration hubs

StrainInfo has always sought to offer users a fully integrated view of available information on microorganisms at the strain level from a variety of sources, while acting as a portal to those very same sources. The main data point on a particular microorganism is, of course, the collection of its strain numbers, one of which the organism would be identified by in publications, in BRC catalogs, on sequence records, and so on. This was the first organising paradigm of

StrainInfo, offering a single page of strain numbers relating to a particular strain, each strain number linked back to the source of information, namely the BRC catalog that offered it. During development of StrainInfo, it soon became apparent that users would benefit from access to further strain related information, the main categories of which are its taxonomic identification, molecular sequence information and publication information, later joined by genome project information, all of which competed for place on this single strain information page.

On the other hand, some of the items available on a StrainInfo passport are interesting objects in their own respect, and a natural entry point for many users in StrainInfo. End users might, for instance, not be aware of the particular strain number of the type strain of a particular species, so it is necessary for them to be able to search StrainInfo using the species name. A particular taxonomic name may also be linked to the various resources related to it, some of which may provide valuable history on the taxonomic name not found directly in StrainInfo. The same could be said about publications or sequences.

Importantly, each of these resources can be cross-linked to other data in StrainInfo, which could not be done in the single page approach. Therefore, StrainInfo has been organised around a unifying principle of 'passport' pages, defined at the strain, taxon, sequence, publication or genome project level. Each passport page includes the main metadata on a particular item, for instance strain numbers on a strain passport. Passports are also the 'integration hub': they feature all data that can be integrated as relating to the object shown on the passport page using available data sources. The main information (such as the taxonomic name assigned to a strain, for instance) are available on the passport itself, but users can explore the available information further by following the links in the cross-reference sections to each information type's own passport. For instance, this means a page such as the strain passport will include a list of molecular sequences derived from the considered strain. Clicking the accession number for a molecular sequence derived from that strain brings up the sequence passport, which in turn links back to the strain passport. Figure 2.1 outlines the different sections of StrainInfo and illustrates how different passports are heavily cross-linked by the various elements listed on each passport page.

In addition to presenting integrated information on data types, the guiding principle of StrainInfo has always been to provide users direct access to the original source of that information. StrainInfo, after all, is merely an aggregator

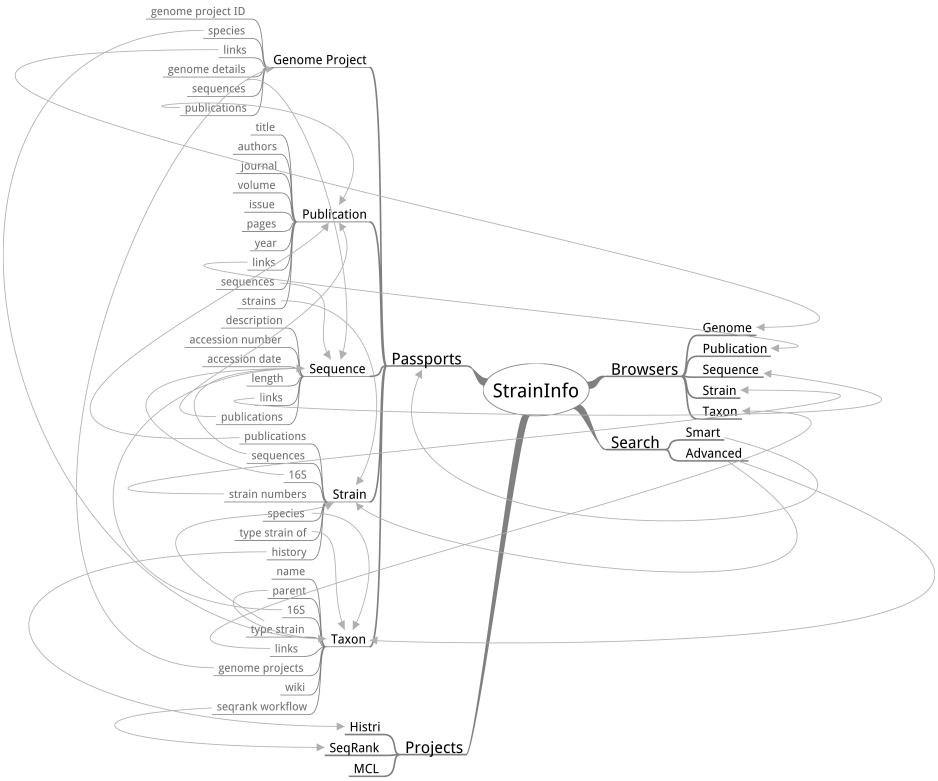


Figure 2.1: Mind map illustrating the layout of the StrainInfo portal and the extensive cross-linking between passports. Each node represents either a page in StrainInfo or an information item on that page. Arrows indicate links from one page to another. Note how passports link mostly to each other, while each also links to a variety of external resources through its respective browser. Smart search allows users to land on a passport page, while advanced search typically presents users with a list of strains that can be followed to the strain passport, or taxon passport since species names are also included in this list.

providing for much improved navigation on strain-related information. It is the original resources where users need to go to perform their actual research tasks, whether that is to order strains or download sequences, acquire publications or examine a genome project's extensive details. Often, there are several, sometimes dozens, such external data sources involved. The best example of this situation is the strain numbers themselves, since different BRCs offer what is essentially a copy of the same strain, along with its metadata. Due to the central role played by BRCs in the distribution of type strains, fourteen percent of type strains have more than twenty known strain numbers. Simply linking to each BRC individually would make it harder to quickly survey the different BRC catalog entries, to easily switch between strain numbers to cross-check information or find better located providers when ordering strains. Therefore, a StrainInfo 'browser' was introduced, which creates an overlay that allows 1-click navigation between several resources external to StrainInfo. The concept can be, and was, naturally extended to other data types, such as molecular sequences and taxonomic information.

Together, passports and browsers make up the main design pattern of the StrainInfo platform, providing a unifying principle not only for user experience, but also programmatic access to the site's data, both internally and externally. By providing useful search functions and heavily cross-linking passports, we ensure that the data is as discoverable to users as possible. The browser function adds to this by making sure users can quickly browse multiple connected objects, without having to cumbersomely click back and forth between the page linking to that. The following sections further explore each passport page and related information.

2.2.2 Strain information

The exact definition of what a strain is tends to vary slightly among authors. In StrainInfo, the definition of strain used is that what [13] calls a "strain in the taxonomic sense", i.e. the set of all descendants of a single pure culture derived from an isolate. Once a culture of a strain has been deposited in a particular BRC, it will be assigned a strain number to track and identify the original deposit. Descendants or subcultures of this culture are typically all referred to by this strain number. When a subculture is moved or transferred to a different BRC, it is assigned a new strain number, which can be used interchangeably with the old strain number. Figure 2.2 illustrates this type of exchange history

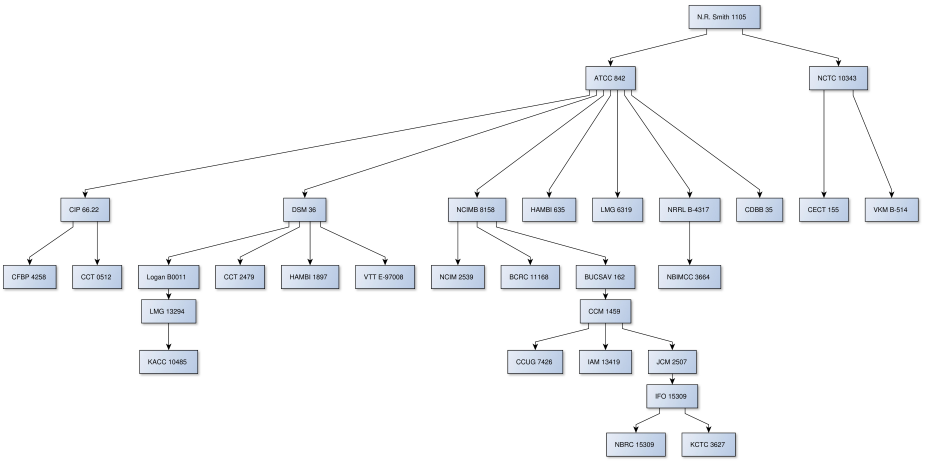


Figure 2.2: Strain numbers and exchange history of the *Paenibacillus polymyxa* type strain. Each node represents a strain number assigned by a BRC or individual researcher, corresponding to what is called a culture in StrainInfo. Each time a culture was subcultured and the resulting culture transfered to a different BRC (indicated by the arrows in this picture), the culture was assigned a new strain number. Each strain number in this graph can be used interchangeably in publications and sequence records to refer to or represent the type strain of *P. polymyxa*, even though they represent different cultures stored in different BRCs. Therefore, StrainInfo describes a strain as the set of all cultures derived from the same original isolate culture, which in this case was assigned the strain number `H.R. Smith 1105`.

for the *Paenibacillus polymyxa* type strain. The strain numbers assigned by different researchers and BRCs, and the biological material they represent, correspond to cultures in StrainInfo, assigned a unique culture identifier inside StrainInfo. In StrainInfo, these individual strain numbers and the biological material they represent are referred to as 'cultures'. In other words, in StrainInfo a culture is the set of all subcultures identified with the same strain number. The strain is the set of all cultures derived from the same original isolate. In Figure 2.2, a single strain is shown, namely the *P. polymyxa* type strain.

Strain number annotations are extremely useful for linking experimental data and publications to the original material. A strain as a whole has no designated name and is in practice always referred to by the strain number of one of

its cultures. When two or more cultures of a particular strain exist, the two are assumed to be clones and their strain numbers can be used interchangeably. It is by this mechanism that some molecular sequences will mention one strain number, while others mention a different one, even though they technically come from the same original microbial material. By defining a strain as the set of all cultures of a strain, both experiments can be reliably tied to the same strain. This approach thus allows for the integration of strain-related data in StrainInfo. The strain passport is the main window into this integration, showing strain information, including all known strain numbers, transfer history (derived in part from BRC culture catalogs), taxonomic data, sequence data and publication data.

The set of strains in StrainInfo is determined algorithmically from BRC culture catalog data and data imports provided by BRCs themselves, described in files formatted using the Microbiological Common Language (MCL) [14], a data standard designed specifically for this purpose. Strain numbers can also be derived from sequence records (when identified via a structured annotation) and other sources when applicable. The integration of the bulk of the strain data is necessarily dependent on the availabity of structured data (in MCL format) from the BRCs themselves, as the only other alternative would be extensive screen scraping of all BRCs. Screen scraping generally consists of downloading and parsing of the HTML web pages of a website. The term screen scraping specifically refers to the fact that the content of those web pages is formatted for consumption by humans and not machines. It usually requires extra effort and hard-to-design heuristics to properly parse this type of information. As there are hundreds of BRCs to be found worldwide, the initial effort to such an endeavour would likely be large but within reach of a small team of developers. The biggest issue to this approach, however, is its fragility in the face of frequent updates to BRC catalogues. Whenever one BRC changes its format, the parser that processes its information needs to be adapted. It is inevitable that most BRCs will make minor changes to their website now and then. Even an assumed average rate of only one change per BRC per year would mean that several parsers need to be updated and re-tested every week. Such an approach simply does not scale, which is why StrainInfo now relies mainly on MCL imports to update strain number information.

Figure 2.3 illustrates how important this integration is, as many strains are distributed very widely, resulting in a plethora of strain numbers being assigned. The most widely distributed strains are understandably type strains, since they

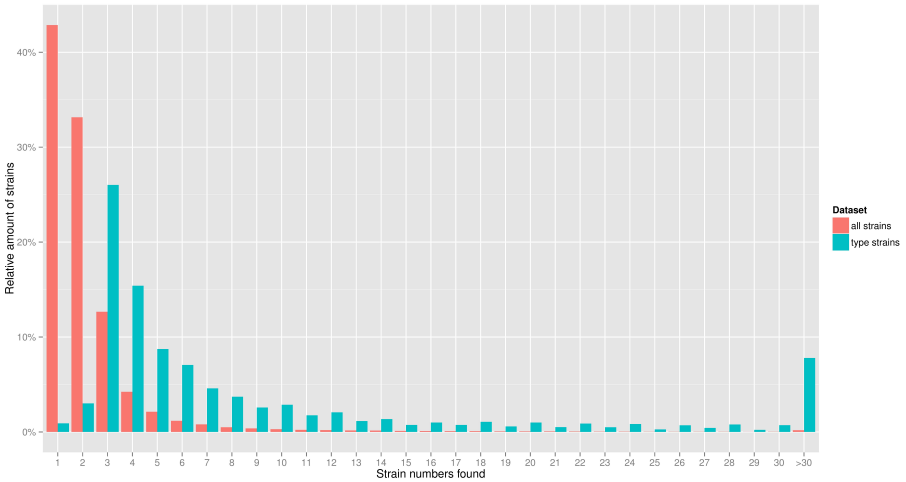


Figure 2.3: Relative distribution of prokaryotic and fungal strains. Type strains are generally deposited in at least two different collections, as required by the rules of new species descriptions, usually resulting in three known strain numbers (a researcher-assigned number, and two BRC-assigned strain numbers).

act as the name bearer of the species and thus are likely more often used and requested than other strains. Most type strains are distributed to at least two BRCs, resulting in three strain numbers being assigned (one researcher number and two collection numbers). A small subset of these type strains are clearly much more widely distributed. This can only partially be explained by the fact that older strains have received more time to be distributed more widely, as this would mean the group would be much bigger. Examining a subset of this more widely distributed group reveals these strains to be generally well studied (model) organisms and/or widely used in economic applications. It can therefore be surmised that the popularity of this core group of 5% to 10% of type strains reflects their current or previous importance in microbial research and applications.

The process by which the StrainInfo integration algorithms determine what cultures belong to the same strain is iterative: as information accumulates and links between cultures become available, or when such links are invalidated, strains are merged, or in the latter case split up. This system was first described

in [10]. As a result of this approach, the composition of a strain, which is a set of cultures in StrainInfo, is often changing, which makes it an unstable entity that cannot be directly identified by a stable identifier, and thus neither can a stable Uniform Resource Identifier (URI) be constructed for these entities. Instead, strains can be identified by their members, the cultures, which are stable entities (corresponding to a single strain number). Cultures do not change over time. Instead, they remain stably identified by their strain numbers, although different collections may use the same strain numbers, especially if they concern research collections that do not assign a unique acronym and number to the strain, such as a BRC would normally do. For this reason, cultures are assigned *culture identifiers*, and each strain passport can be accessed and referenced the culture identifier of any of its cultures. While the composition of that strain passport might change (for instance when adding more cultures), the URI of the page, by including the culture ID, always links to the passport of the strain represented by this particular culture.

The strain passport is subdivided in several sections, namely:

- The overview section, containing strain numbers and links to general metadata of the strain.
- An image of the Histri [15] of this strain, which is a depiction of the exchange history of this strain. Each node in a histri figure corresponds to a set of strain numbers which actually refer to the same culture, but have been assigned different culture identifiers because they could not practically be distinguished; usually this concerns different spellings of the same strain number. A link between two nodes indicates a transfer of this strain from one institute or researcher to another.
- A list of molecular sequences derived from the strain. This list is constructed by examining taxonomic resources, strain catalogs and molecular sequence records extracted from the INSDC databases (through the EBI ENA database).
- A list of publications the strain was involved in as biological material, derived from the same resources as the list of molecular sequences.

Figure 2.4 shows a typical overview section of the strain passport. Shown in the page heading on every strain passport is the strain number and the species name found for this strain number. The overview section itself contains all

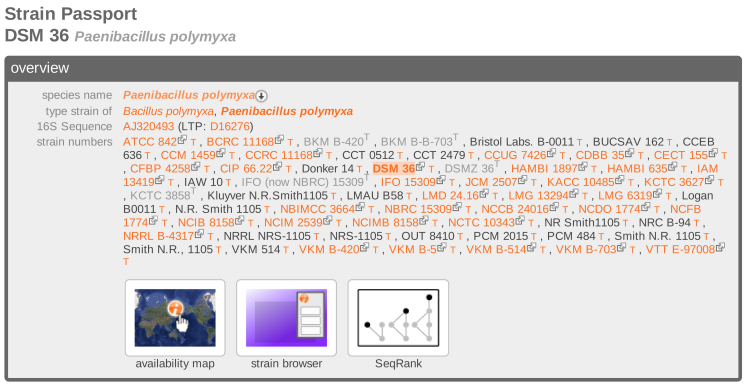


Figure 2.4: Overview section of the strain passport, from the perspective of *P. polymyxa* type strain culture DSM 36^T. This culture has been assigned culture ID 39157, and the corresponding full strain passport can be accessed at `http://www.straininfo.net/strains/39157`.

strain numbers. Clicking through on the linked strain numbers transports the user to a strain browser, which loads the culture catalog page associated with a particular strain, next to a list of equivalent strain numbers that can be used to quickly navigate to the catalog pages of those strain numbers. Its main function is to enable quick comparison between multiple catalogs, for instance to check taxonomic details or for comparative shopping before ordering a strain. In addition to strain numbers, the overview section shows type strain information, all species names by which individual strain numbers have been identified, and the accession number of a representative 16S rRNA gene sequence determined by StrainInfo. A few large buttons on the bottom of the window provide easy access to a map of the availability of this strain worldwide, obtained by mapping strain numbers to the location of the BRC that issued them, to the strain browser for the current culture, and to a ranked list of 16S rRNA gene sequences of this strain.

Further down on the strain passport the histri is included, an example of which can be seen in Figure 2.5. As explained above, users arrive at the strain passport from the perspective of a particular culture or strain number. This culture is therefore highlighted (with orange color) in the histri figure, which allows it to be straightforwardly found in what can be a large and somewhat complicated exchange history.

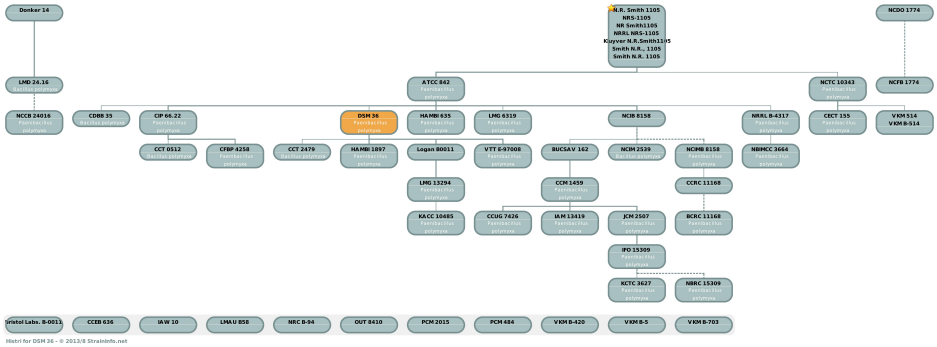


Figure 2.5: Histri figure of the *P. polymyxa* type strain DSM 36^{T}, as extracted directly from its strain passport at time of writing.

The final two sections contain a list of all molecular sequences and a list of all publications known to StrainInfo for this strain, each of which shows both some general crucial information (for instance sequence length), while also linking to the respective passport. In this way, as much basic and useful information can be shown on the same page, while extra data can be shown on individual passports pages when users choose to explore the data further. The same relationship holds for all other information on this page that can be linked to a passport, such as taxonomic names, except for strain numbers in the overview list, which link directly to the strain browser.

2.2.3 Taxonomy

Biological taxonomy seeks to classify all organisms in taxonomic groups, whose hierarchical relationship express a classification on the basis of similarity between members of each group [16]. Names are organised in taxonomic groups at different ranks in the taxonomic tree of life. In principle, there are seven ranks: kingdom, phylum, class, order, family, genus and species. Some of these ranks also have subranks, e.g. orders can have suborders, species can have subspecies. Above the rank of kingdom, the rank of domain is also commonly used, according to the three domains, Bacteria, Archaea and Eucarya (also referred to as Eukarya or Eukaryota), proposed by [4]. Names of Bacteria and Archaea up to the level of class are governed by the Bacteriological Code [17], while those of Fungi are in principle governed by the Melbourne Code [18]. Opinion on

the organisation of the higher taxonomies in particular can vary greatly, but both codes have in common that individual strains are uniquely classified by a two-name combination of genus and species, optionally involving subgenus and subspecies in certain situations. It is this name that is linked on strain passports and elsewhere in StrainInfo and which typically leads to a taxon passport at the species rank. Taxonomy for Fungi is restricted to the species and genus level, while for Bacteria and Archaea the full taxonomy according to the List of Prokaryotic Names with Standing in Nomenclature (LPSN) is also available, each name at each rank having been assigned its own taxon passport.

The taxon passport

Each taxon listed in StrainInfo is assigned its own identifier and its own taxon passport. Whenever this name is used in StrainInfo then, it is linked to this taxon passport, which in this way has become a hub for exploration of information on the passport it describes. Figure 2.6 depicts the taxon passport of a species, *Paenibacillus polymyxa*. Taxon passports can represent a taxon at any level of the taxonomic tree: from the lowest level (i.e. subspecies) to the highest (i.e. domain). In addition to listing taxon name and rank, this page can also be used to find the taxon of the immediately higher rank that includes the taxon on the current passport (following Computer Science nomenclature developed for tree data structures, this is referred to as the 'parent taxon') and all taxa included in this taxon, if any, at immediately lower rank. Following the same naming scheme, these can be referred to as the 'children' of the taxon, although in StrainInfo they are called 'subtaxa'. Providing these links makes it much easier to navigate around taxonomic nomenclature, which can be used in conjunction with the related publications feature to quickly gather information on the parts of the taxonomy users are often interested in. Links are included to external resources, which often contain additional information on taxa and are themselves data sources for some of the information on this passport. There are additionally several efforts to create collaboratively edited pages describing all known species, included here as separate links in the "Wiki" category.

The type strain forms the cornerstone of bacterial taxonomy, acting as the name bearer and representative strain of a species, and is thus pivotal for identification purposes. It is displayed prominently on this page, listing all strain numbers with the option to access the type strain passport for further information. Two links are also included to launch searches for related strains

Taxon Passport
Paenibacillus polymyxa

overview

species	*Paenibacillus polymyxa*
parent taxon	*Paenibacillus sp.*
type strain	ATCC 842 T, BCRC 11168 T, BKM B-420^T, BKM B-B-703^T, Bristol Labs. B-0011 T, BUCSAV 162 T, CCEB 636 T, CCM 1459 T, CCRC 11168 T, CCT 0512 T, CCT 2479 T, CCUG 7426 T, CDBB 35 T, CECT 155 T, CFBP 4258 T, CIP 66.22 T, Donker 14 T, DSM 36 T, DSMZ 36^T, HAMBI 1897 T, HAMBI 635 T, IAM 13419 T, IAW 10 T, IFO (now NBRC) 15309^T, IFO 15309 T, JCM 2507 T, KACC 10485 T, KCTC 3627 T, KCTC 3858^T, Kluyver N.R.Smith1105 T, LMAU B58 T, LMD 24.16 T, LMG 13294 T, LMG 6319 T, Logan B0011 T, N.R. Smith 1105 T, NBIMCC 3664 T, NBRC 15309 T, NCCB 24016 T, NCDO 1774 T, NCFB 1774 T, NCIB 8158 T, NCIM 2539 T, NCIMB 8158 T, NCTC 10343 T, NR Smith1105 T, NRC B-94 T, NRRL B-4317 T, NRRL NRS-1105 T, NRS-1105 T, OUT 8410 T, PCM 2015 T, PCM 484 T, Smith N.R. 1105 T, Smith N.R., 1105 T, VKM 514 T, VKM B-420 T, VKM B-5 T, VKM B-514 T, VKM B-703 T, VTT E-97008 T
16S rRNA gene	AJ320493 (LTP: D16276)
external links	Catalogue of Life, DSMZ, LPSN; J.P. Euzéby, NCBI, WikiSpecies
Wiki	Wikipedia Article
search StrainInfo	Find all strains Find all type strains

SeqRank workflow

publications

4 items found, displaying all items.

Amoozegar MA, Malekzadeh F, Malik KA, Schumann P, Sproer C
Halobacillus karajensis sp. nov., a novel moderate halophile
Int J Syst Evol Microbiol 53(4), 1059-1063, 2003

Paenibacillus durus (Collins et al. 1994, formerly Clostridium durum Smith and Cato 1974) has priority over Paenibacillus azotofixans (Seldin et al. 1984). Opinion 73
Int J Syst Evol Microbiol 53(3), 931, 2003

Validation of the publication of new names and new combinations previously effectively published outside the IJSB. List No. 51
Int J Syst Bacteriol 44, 852, 1994

Buchanan, RE, Gibbons, NE
(Unknown)
Bergey's manual of determinative Bacteriology 8th ed., 1974

4 items found, displaying all items.

Figure 2.6: Taxon passport for the species *Paenibacillus polymyxa*. The passport contains useful information on the species such as the identity of the type strain and a representative 16S rRNA gene sequence, and links to external entities via the taxon browser, an example of which is shown in Figure 2.7. The SeqRank logo can be clicked to access the SeqRank workflow of this taxon (examined in detail in Chapter 6). A list of publications is also provided, with each entry linking to the corresponding publication passport. Navigation links to parent taxon and subtaxa (if any) allow for navigation of the entire taxonomic tree as extracted by LPSN.

(e.g. strains that contain cultures identified as being part of this taxon). Next to strain information, users are often in need of finding a representative high-quality molecular sequence for use in phylogeny. A representative 16S rRNA gene sequence is therefore also given on this page. This sequence is selected automatically by the StrainInfo SeqRank algorithm. The underlying algorithm and extended workflow linked to from this page are explained in more detail in Chapters 5 and 6. In future, most species will have the entire genome of at least one of their strains fully sequenced, as initiatives have sprung up to sequence the genome of all available type strains [19, 20]. A sizable fraction already have projects to do just that underway or entirely finished, available on their own genome project passports in StrainInfo, linked here on the taxon passport where available.

Usually, the best source of information on a taxon is still its original publication. Only these publications which appear to directly describe a taxon are included on this page, as usual cross-linking to their publication passports. Users of course can click through to the publication lists of the type and other strains to find more. In addition, external curated taxonomic databases are displayed prominently via the aforementioned links section, often providing a very good place to start for literature searches in the taxonomy.

Taxonomic information sources

While the bacteriological code provides specific rules governing the nomenclature of new and existing species, the correct name and classification for any particular strain is subject to expert opinion. Likewise, in the world of fungi many strains have quite a lot of different, often equally valid names. The problem is compounded by the way taxonomic changes are handled, which causes species names to regularly be reassigned to new genera, strains to be reassigned to new species, genera to be reassigned to different families, and so on. As a result, the taxonomic name and type strain information listed for a certain culture in any one of the culture collections, while the first source of information, is not always up-to-date and reliable. It is for this reason that StrainInfo displays such found names on the strain passport, but also indexes and links to several external, curated data sources, in order to have access to an up-to-date and reliable view of the taxonomy. Different authors still may disagree, which is handled by providing links to a healthy selection of them and standardising on certain taxonomies when it comes to crucial information,

such as type strain status.

Available taxonomies are generally either fully manually curated, or in part constructed automatically (or through user submissions) from external metadata, such as experimental molecular sequence data annotations. In all cases authors stress that theirs is simply a curated list of names considered valid, but that the status of the taxonomy can only be ascertained fully by an expert in the field. While in principle true, most of these lists can and are in practice used as a taxonomy, as they are in StrainInfo to classify strains.

The source of the type strain information for Bacteria and Archaea in StrainInfo, is the nomenclature list first published by J.P. Euzeby (retired) and now curated by Aidan C. Parte, the so-called List of Prokaryotic Names with Standing in Nomenclature (LPSN) [11]. At time of writing it can be found online at `http://www.bacterio.net`. It is based on careful consideration of the publication record, and type strain information is kept meticulously up to date. Unfortunately it is not available in an easy-to-parse electronic format, but the information that can be gleaned automatically has been added to StrainInfo, while further data such as publication background can be found on the website itself through the taxon browser, shown in Figure 2.7. The equivalent data source for fungi is Mycobank, which indexes mycological names and organism information [21].

Other taxonomies exist; often they differ subtly in what names they choose to include or are much more inclusive than any listing including only validly published names. A first notable example of these included in StrainInfo is the NCBI taxonomy [22], which is a semi-curated taxonomy that is also very inclusive, since it include names suggested by molecular sequence submitters, even before they have been validly published (if ever). It is often used for the generation of automated phylogenetic pipelines. A stricter taxonomy is provided by the Prokaryotic Nomenclature Up-to-date published by DSMZ[1], which is curated in a similar manner as the LPSN and differs slightly in the inclusion or exclusion of several species based on expert opinion, and in the particular spelling of entries. Links to these taxonomies are included on all taxon passports for which a link could be found.

[1] The Prokaryotic Nomenclature Up-to-date can be found at `http://www.dsmz.de/bacterial-diversity/prokaryotic-nomenclature-up-to-date.html`

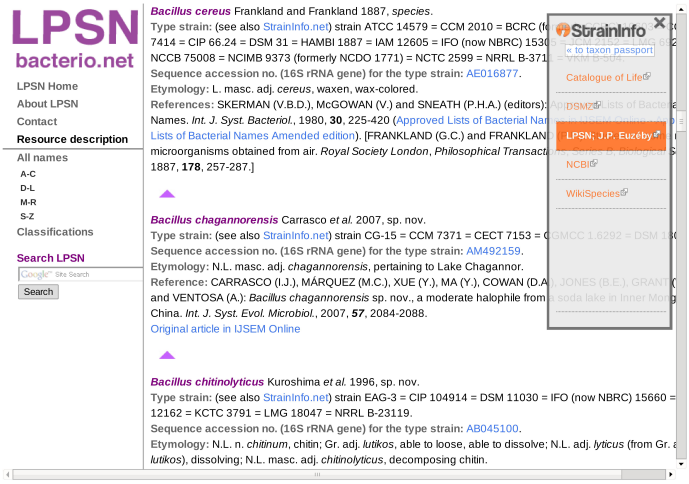


Figure 2.7: Taxon browser opened on the LPSN entry for *Bacillus cereus*. The underlying frame loads the original entry on the website, while the StrainInfo overlay shown on the right enables switching between other entries for this taxon at known taxonomic data sources.

2.2.4 Molecular sequence data

Taxonomy is in principle only concerned with the classification of strains. Originally, this could only be done on the basis of experimentally observed physical properties of strains under a light microscope, soon expanded with a wide array of biochemical tests used to classify strains in different groups of similar species. The modern microbiological species definition, however, has increasingly been based on genetic similarity [13], to the point where much of taxonomy is now largely informed by phylogenetic methods. While the gold-standard test of relatedness between strains to determine species remains DNA-DNA hybridisation, several marker genes were eventually found that provide enough information to distinguish strains at least at the genus level, most notable of which is the 16S rRNA gene, which in a large number of genera is sufficiently different between different species to delineate them [3, 23–25]. Where more resolution is needed, researchers have taken to analysing multiple genes at once, in multilocus sequence analysis (MLSA) schemes, or even whole-genome sequencing. Identification methods have shown a similar trend: initially relying

on biochemical tests, laboratories now routinely incorporate sequencing in their workflow at ever decreasing cost. The same data is of course also used in functional genomics and other basic research fields, such as microbial ecology. As the cost of genome sequencing has fallen, it is increasingly easy to sequence the whole genome of strains under consideration, with the only constraints now being the analysis and storage of these molecular sequences [26].

Initially, molecular sequences where simply published as-is in paper form. With the advent of better sequencing methods and ever more, and longer sequences, this situation was not tenable for long. Since the sequence data itself is ultimately necessary for the practical reproducibility of research, it stands to reason that it should be available from a central and stable location wherever possible. The International Sequence Database Collaboration or INSDC has been providing this service for more than twenty years now [6] and its data store continues to increase. Member databases provide a central location into which to deposit newly generated sequences, and assign them a globally unique identifier called an accession number, which can be used to refer to the sequence in print and in sequence analysis pipelines. All sequences currently curated in these databases can be individually accessed online, and are also exported wholesale in large database releases which can be downloaded. StrainInfo relies in this case on a combination of the latest release from the European Nucleotide Archive (ENA) maintained by the European Bioinformatics Center (EBI), and the files with daily sequence updates that the EBI provides, keeping the internal sequence information database used by StrainInfo synchronized with what is publicly available.

Sequence passport

The sequence passport, an example of which is shown in Figure 2.8, is comparatively simple, showing simply the accession number and description of the sequence and accession date and length. When the sequence can be linked to a strain, the strain number of the specific culture it is linked to is displayed and selecting it leads the user to the relevant strain passport. Likewise, possible publications in which the sequence was mentioned are displayed and linked to a publication passport. Aside from the length, more details about the sequence (including the sequence itself) can be found in one of the three INSDC databases, namely Genbank [27], the European Nucleotide Archive (ENA) [28] and the DNA Databank of Japan (DDBJ) [29]. They are linked here using a

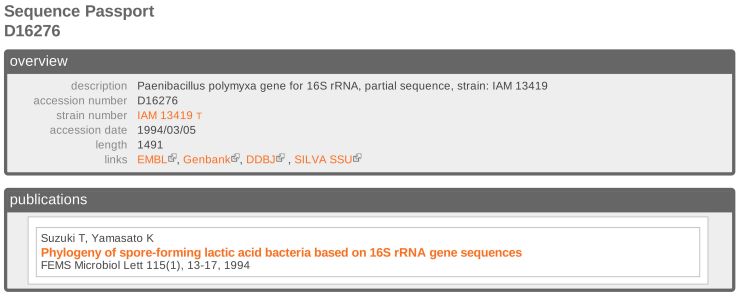


Figure 2.8: Contents of the sequence passport for the sequence record with accession number D16276. The sequence passport links to external INSDC resources using the same browser concept found on other passports. Other passports are linked through the strain number of the source material, which links to a strain passport, and a list of publications retrieved from the sequence record in ENA. Each entry in the publication list links to a separate publication passport, which further links to the original publication if such a link can be established.

sequence browser. While each contains similar information, they do so in different formats. The sequence browser can be used to switch between these representations. When the sequence is determined to be of a 16S or 23S rRNA gene sequence, a link is also included to its record in the SILVA database [30, 31], which specialises in analysis of small and large subunit rRNA gene sequences.

2.2.5 Publication passports

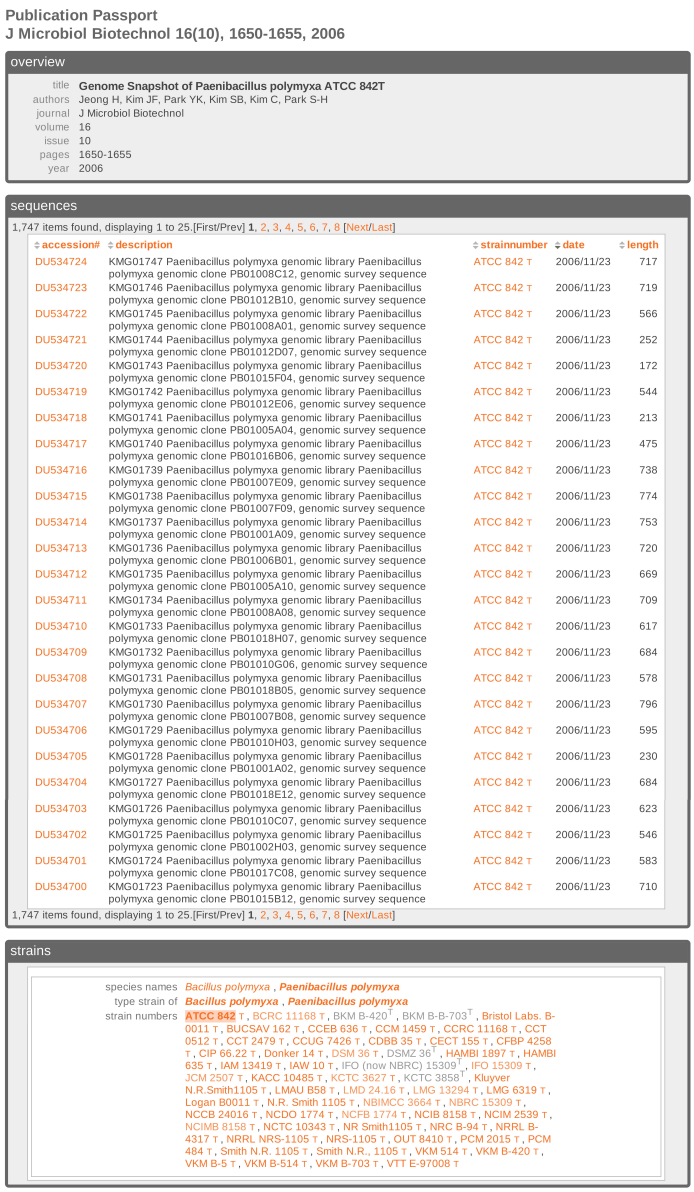
Publication passports provide basic bibliographic information on the publication, such as title, date of publication and authors. Since sequence records often include a reference to the publication the sequence was generated for, this provides a natural method to obtain a full listing of molecular sequences generated for a particular publication, provided that the authors of the publication have made sure to properly annotate the records in INSDC. By following the links thus created, it is also possible to display related strains, e.g. source material or the type strain of the described species, on the same page. As usual these sections also cross-link to the respective sequence and strain passports. The publication itself can often be found from several different providers, via identifiers such as the DOI or through Pubmed, both of which are linked as

usual to a publication browser.

The example of Figure 2.9 shows the value of supplying a separate passport for each publication. The publication shown here concerns a large genome survey of *Paenibacillus polymyxa* ATTC 842^{T}. Each sequence record involved in this survey refers to the publication it was involved in, a link to which is shown on each related sequence record. Using this link, the full set of sequences can be displayed on the strain passport. Since the sequence records also contain the strain number of the source strain, it is possible to also include the strain involved. When the sequence records linked to a publication link to more than one strain, those other strains are shown too. This shows that by traversing the links available, the value of a single passport can be quickly augmented with interesting and related information. Should the user require more details about one of the sequence or strain, he or she can then navigate to the passport of his or her object of interest, on which more information and links to source information can always be found.

2.2.6 Genome passport

Large-scale sequencing projects tend to result in a variety of submissions to the INSDC. For a whole-genome sequencing project, for instance, initial short sequence reads might be submitted, followed by a draft genome and later by a finished genome sequence. Much metadata is attached to these sequences and in general the project itself. NCBI bioprojects serve as a repository for this data and a way to tie the metadata together with the large variety of various submissions [32]. In the Genomic Rosetta Stone (GRS) Project, a project undertaken under the auspices of the Genomic Standards Consortium, infrastructure was developed to link genome (project) records to additional analyses from a wide array of third party data providers. The genome project passport provides basic data on bioprojects extracted from the bioproject data exports, and provides a window into the GRS data, by allowing users to browse the various data resources that have registered their data records within GRS. The current version of GRS was implemented and released as part of this work, and is described in greater detail in Chapter 4.



Publication Passport
J Microbiol Biotechnol 16(10), 1650-1655, 2006

overview

title	**Genome Snapshot of Paenibacillus polymyxa ATCC 842T**
authors	Jeong H, Kim JF, Park YK, Kim SB, Kim C, Park S-H
journal	J Microbiol Biotechnol
volume	16
issue	10
pages	1650-1655
year	2006

sequences

1,747 items found, displaying 1 to 25.[First/Prev] **1**, 2, 3, 4, 5, 6, 7, 8 [Next/Last]

accession#	description	strainnumber	date	length
DU534724	KMG01747 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01008C12, genomic survey sequence	ATCC 842 T	2006/11/23	717
DU534723	KMG01746 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01012B10, genomic survey sequence	ATCC 842 T	2006/11/23	719
DU534722	KMG01745 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01008A01, genomic survey sequence	ATCC 842 T	2006/11/23	566
DU534721	KMG01744 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01012D07, genomic survey sequence	ATCC 842 T	2006/11/23	252
DU534720	KMG01743 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01015F04, genomic survey sequence	ATCC 842 T	2006/11/23	172
DU534719	KMG01742 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01012E06, genomic survey sequence	ATCC 842 T	2006/11/23	544
DU534718	KMG01741 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01005A04, genomic survey sequence	ATCC 842 T	2006/11/23	213
DU534717	KMG01740 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01016B06, genomic survey sequence	ATCC 842 T	2006/11/23	475
DU534716	KMG01739 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01007E09, genomic survey sequence	ATCC 842 T	2006/11/23	738
DU534715	KMG01738 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01007F09, genomic survey sequence	ATCC 842 T	2006/11/23	774
DU534714	KMG01737 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01001A09, genomic survey sequence	ATCC 842 T	2006/11/23	753
DU534713	KMG01736 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01006B01, genomic survey sequence	ATCC 842 T	2006/11/23	720
DU534712	KMG01735 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01005A10, genomic survey sequence	ATCC 842 T	2006/11/23	669
DU534711	KMG01734 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01008A08, genomic survey sequence	ATCC 842 T	2006/11/23	709
DU534710	KMG01733 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01018H07, genomic survey sequence	ATCC 842 T	2006/11/23	617
DU534709	KMG01732 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01010G06, genomic survey sequence	ATCC 842 T	2006/11/23	684
DU534708	KMG01731 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01018B05, genomic survey sequence	ATCC 842 T	2006/11/23	578
DU534707	KMG01730 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01007B08, genomic survey sequence	ATCC 842 T	2006/11/23	796
DU534706	KMG01729 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01010H03, genomic survey sequence	ATCC 842 T	2006/11/23	595
DU534705	KMG01728 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01001A02, genomic survey sequence	ATCC 842 T	2006/11/23	230
DU534704	KMG01727 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01018E12, genomic survey sequence	ATCC 842 T	2006/11/23	684
DU534703	KMG01726 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01010C07, genomic survey sequence	ATCC 842 T	2006/11/23	623
DU534702	KMG01725 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01002H03, genomic survey sequence	ATCC 842 T	2006/11/23	546
DU534701	KMG01724 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01017C08, genomic survey sequence	ATCC 842 T	2006/11/23	583
DU534700	KMG01723 Paenibacillus polymyxa genomic library Paenibacillus polymyxa genomic clone PB01015B12, genomic survey sequence	ATCC 842 T	2006/11/23	710

1,747 items found, displaying 1 to 25.[First/Prev] **1**, 2, 3, 4, 5, 6, 7, 8 [Next/Last]

strains

species names *Bacillus polymyxa* , ***Paenibacillus polymyxa***
type strain of ***Bacillus polymyxa*** , ***Paenibacillus polymyxa***
strain numbers ATCC 842 T , BCRC 11168 T , BKM B-420^{T} , BKM B-B-703^{T} , Bristol Labs. B-0011 T , BUCSAV 162 T , CCEB 636 T , CCM 1459 T , CCRC 11168 T , CCT 0512 T , CCT 2479 T , CCUG 7426 T , CDBB 35 T , CECT 155 T , CFBP 4258 T , CIP 66.22 T , Donker 14 T , DSM 36 T , DSMZ 36^{T} , HAMBI 1897 T , HAMBI 635 T , IAM 13419 T , IAW 10 T , IFO (now NBRC) 15309^{T} , IFO 15309 T , JCM 2507 T , KACC 10485 T , KCTC 3627 T , KCTC 3858^{T} , Kluyver N.R.Smith1105 T , LMAU B58 T , LMD 24.16 T , LMG 13294 T , LMG 6319 T , Logan B0011 T , N.R. Smith 1105 T , NBIMCC 3664 T , NBRC 15309 T , NCCB 24016 T , NCDO 1774 T , NCFB 1774 T , NCIB 8158 T , NCIM 2539 T , NCIMB 8158 T , NCTC 10343 T , NR Smith1105 T , NRC B-94 T , NRRL B-4317 T , NRRL NRS-1105 T , NRS-1105 T , OUT 8410 T , PCM 2015 T , PCM 484 T , Smith N.R. 1105 T , Smith N.R., 1105 T , VKM 514 T , VKM B-420 T , VKM B-5 T , VKM B-514 T , VKM B-703 T , VTT E-97008 T

Figure 2.9: Publication passport of a publication concerning a genome survey of the genome of *Paenibacillus polymyxa* ATTC 842^{T}. By extracting the publication from sequence records, StrainInfo can display all sequence records from the survey on this page, as well as the strain numbers of the relevant strain.

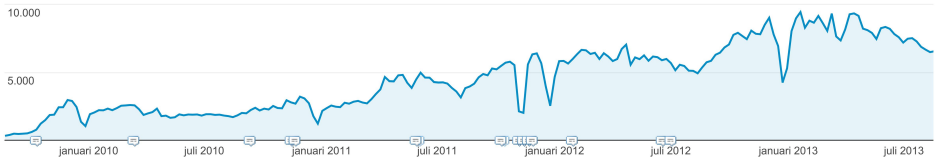


Figure 2.10: Weekly user numbers for StrainInfo from Aug 23, 2009 (when measuring started) until Aug 23, 2013. The site has steadily gained popularity over the years, from 327 visits in the first week of measurements, to 6,549 visitors in the last week included in this picture.

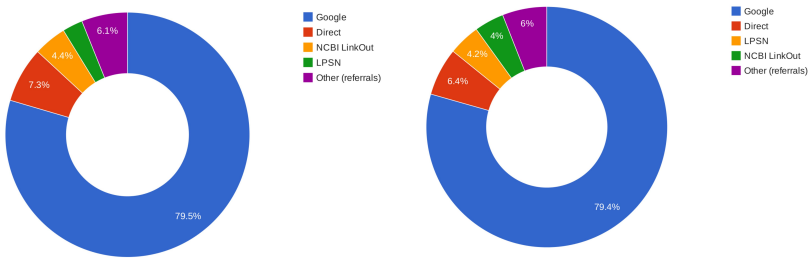
2.3 Finding passports: search and referrals

Two search functions allow the user to locate passports: Smart search and Advanced search. Smart search is the search function performed whenever a user enters some search query in the search box included on most StrainInfo pages. It iteratively searches the various passports for the information the user is searching for, starting with strains, then trying taxa, molecular sequences and so on. In this way, if a user enters a particular strain number, or the name of a species, he or she will be directed straight to its passport page. If no passport specific to the query was found anywhere in StrainInfo, Smart search performs a normal strain search with the query string the user entered as its input. Advanced search is the means by which such a regular strain search can be further narrowed down to include only type strains, strains from a certain taxon, strains linked to a particular molecular sequence or strains matching a full text query in StrainInfo's legacy database. Together they form the basic portal to information on StrainInfo for visitors who visit the site directly.

The two main search functions are specialised in that they search directly on record fields. Sometimes it may be more useful to have access to full-text search of an entire passport page. For this reason, an index is provided to external search engines to all pages in StrainInfo in the form of a site map, following the Sitemap protocol[2]. External search engines such as Google use generated files linked to by a root `sitemap.xml` file[3] following this protocol to find all pages on StrainInfo, and index them accordingly.

StrainInfo as a website has steadily gained users over the years, starting from

[2]The Sitemap protocol is described in detail at `http://www.sitemaps.org/protocol.html`
[3]Found on StrainInfo at `http://www.straininfo.net/sitemap.xml`



(a) All visitors **(b)** Returning visitors

Figure 2.11: Origin of all visitors (a) and returning visitors (b) to StrainInfo for the year 2012. The individual visitor source is shown if it accounts for more than 1% of visits. The large majority of users arrive at StrainInfo through Google (>79% in both cases). Many also access StrainInfo directly, for instance through bookmarks. In a number of cases, users reach StrainInfo through other referring websites or smaller search engines. The two single biggest sources of traffic in this category are, by far, LPSN and the NCBI LinkOut system.

328 users in the first week since measurements were taken, to an average of 7,700 users in the last year. Figure 2.11 graphs the weekly user numbers over time, showing a long but steady rise and fluctuations during vacation periods, especially the Christmas period and a drop in visitors in the summer months. As the last week included occurs during the summer month of August, the number of users is somewhat lower than the average, at 6,643.

Figure 2.11 shows by which means users arrived at the StrainInfo website in the year 2012. The overwhelming majority of users are referred to content on StrainInfo by a search engine. The figure clearly shows how dominant Google is as a search engine, with no other search provider directing more than 1% of users to StrainInfo. What is interesting is that not only new visitors find the site through Google, but also returning visitors. This underlines the importance of fully indexing the site in external search engines, as it is the preferred access method of a large section of our audience. As well as linking to external resources, StrainInfo is also linked as a strain information resource from several other sites, in particular the List of Prokaryotic Names with Standing in Nomenclature (LPSN) [11] and molecular sequence information pages on the NCBI website, through the NCBI LinkOut system [33]. Both methods appear

to be a valuable way for users to discover strain passports, since they together account for more users than the number that directly access StrainInfo or use its search function.

2.4 StrainInfo as a software platform

The current version of StrainInfo, referred to internally as StrainInfo 2.0, is built up from the ground up as a modular architecture, organised in the form of a set of modules defined by the Maven build tool[4] which bundle the Java code comprising a large part of the web platform. In addition to this, legacy functionality implemented in-database as PL/SQL stored procedures provides crucial parts of data integration and extraction functionality. Figure 2.12 contains an overview of the various modules and how they interact with both the data store and users. Users include not only end users, but also the developers themselves and automated programs that leverage the StrainInfo infrastructure to perform important tasks such as updating cache data or resetting user passwords. Each of the modules listed in Figure 2.12 can be built separately, and dependencies between modules are defined explicitly in the configuration files. Most of the StrainInfo website, including the core library module, is implemented in the Java programming language and linked dynamically using Maven. The Spring dependency injection framework[5] links individual classes together at runtime.

As the name suggests, StrainInfo 2.0 replaces an older version of StrainInfo, and thus interacts with several legacy programs and data schemes integrated into an Oracle Relational Database Management System (RDBMS). As is typical, this has sometimes led to some compromises when it comes to data modeling, since such a legacy design cannot always easily be modified. The `core` module contains this problem by concentrating all data access related functionality into a single data access layer, implemented with the Hibernate Object-Relational Mapping (ORM) framework[6]. In addition, much of the StrainInfo logic, such as that for parsing strain numbers, should be shared among modules. All such functionality is built as separate packages within the core module. All data access within the core module or externally, however, always happens through the data access layer defined in this core module. This design can

[4] `http://maven.apache.org/`
[5] `http://www.springsource.org/spring-framework`
[6] `http://www.hibernate.org`

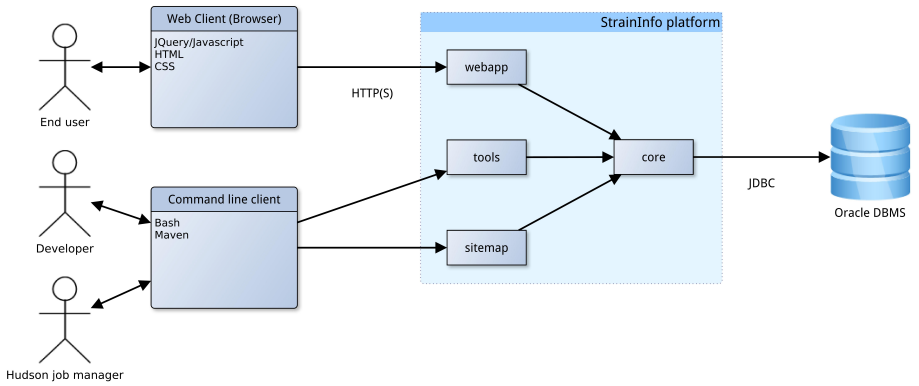


Figure 2.12: Diagram of the most important modules and interactions involved in the StrainInfo platform architecture. Rectangle nodes indicate software modules and libraries. Arrows indicate direction of dependencies or communication channels.

be cumbersome when a change affects multiple layers, but it is much more common that slight tweaks are necessary in the way the application interacts with its data store, which can then be localised within only this module.

No other module connects directly to the database, and all modules typically share a dependency on the Struts framework. The `webapp`, `tools` and `sitemap` modules provide different types of access and specialised functionality to StrainInfo. The `webapp` module contains StrainInfo as most people know it: the website and assorted services, implemented as a Struts 2[7] web application. The `tools` and `sitemap` modules have different users, namely the developers of StrainInfo and StrainInfo's Hudson Continuous Integration Server. Developer can use it to run once-off analyses or reset a user password, for instance, while Hudson has been extended to periodically call certain tasks, as is necessary to create data exports or the sitemap used by external data engines (and built by the `sitemap` module). The complete extent of this background automation is explored further in Chapter 3.

[7] `http://struts.apache.org/development/2.x/`

2.5 Programmatic access points into StrainInfo

StrainInfo is part of a large ecosystem of services aimed at supporting the practice of microbiology, in the sense that it is a growing and changing member filling its niche in this space. This analogy also applies to its interaction with other services, since StrainInfo is dependent relies on outside data sources for data integration, while in the same way outside sources often wish to process strain-related information automatically for their own purpose. In addition, workflow tools such as Kepler [34] and Taverna [35] allow advanced users to programmatically access bioinformatics services in a straightforward way. Other users may be more comfortable with scripted access to the same data using general purpose programming languages. In all cases, the availability of machine readable data (as opposed to data formatted for display to human users) and specific web services greatly reduces the effort involved in accessing StrainInfo data in an automated way.

2.5.1 SOAP web services

A wide range of automation frameworks such as the aforementioned workflow tools rely on the availability of web services that support the Simple Object Access Protocol (SOAP) [36]. This type of web service definition takes a procedural approach to internet data messaging, and works by publishing a set of operations supported by a service, as well as the message format needed to communicate with it. In StrainInfo, the java JAX-WS framework is used to implement a select set of services to support finding a culture ID, given a strain number. The first of these, the `Resolver` service will only return a result if the strain number provided can be unambiguously resolved to a culture ID. The second, the `Search` service, will return multiple results when it cannot uniquely resolve a strain number, and can also be used in conjunction with a species name, which can help narrow down the strain number to a unique culture. Table 2.1 lists the different services and their operations.

2.5.2 Direct access using RESTful like URI

Providing an API can also be done by reusing the same URIs and layout as is used for the website proper, if sufficiently carefully designed. This type of API design is loosely based on the Representational State Transfer (REST)

Service	Operation	Input	Output
Resolver	resolveCulture	strain number string	culture ID
	resolveCultures	list of strain numbers	list of culture IDs, one per strain number
Search	searchCulture	strain number	list of culture IDs
	resolveCultureOfSpecies	strain number, species name	culture ID

Table 2.1: List of SOAP web services and their operations.

paradigm described by [37], a style of design specific to the Web which relies on a particular set of operations being available and a uniform interface and data type exposed by the web service, which can be made amenable to automation by providing machine readable representations. In StrainInfo, many of the principles of this design style are followed. However, a RESTful design requires in principle that the entire structure of the interface is available as part of that interface. For example, plain HTML pages themselves are RESTful in the sense that clients need only understand the concept of URIs and links in order to navigate and use the application (in this case a website). In contrast, some of the features exposed by StrainInfo, such as a machine readable version of the strain passport, are not included directly in the default representation of the website. Different media types are employed for the user readable and machine readable version of the site, which further contrasts with the REST style of design. This approach makes it easier to evolve the graphical design and features of the website, while still retaining some of the attractive features of REST, the most importance of which to users are a data-oriented interface and URI scheme, and the possibility of programmatic access using simple HTTP clients, of which many implementations are readily available.

Navigability of the site and access to machine readable versions of the data is accomplished using a fixed set of conventions, namely:

- Every strain passport provides a machine readable version, accessed by appending the extensions '.xml' to any strain passport URI, e.g. the machine readable version of `http://www.straininfo.net/strains/4490` can be found at the URI `http://www.straininfo.net/strains/4490.xml`.

- All passports and browsers in the site are divided into uniform namespace, where each set of objects has its own namespace. Each object

within a namespace must be uniquely identified by a single, and stable identifier.

- Any namespace can be searched with multiple possible identifiers by using a specialised namespace search.

Developers of clients to the StrainInfo platform may rely on the fact that these rules will always be enforced within the main passport namespaces, allowing them to easily build clients without fear of them being obsoleted in future versions. This is further reinforced by choosing stable data representations, such as the MCL standard for the machine readable version of the strain passport.

When the goal of clients is simply to link to StrainInfo, they can rest assured that identifiers (and the related URIs) won't change and break the links they have assigned, which is a common problem on the web. By providing unique identifiers for cultures that never changed, this has also introduced a way in which strain numbers can be deduplicated. Strain numbers after all are not always globally unique between different collections, but the culture IDs they are assigned within StrainInfo are. Therefore, the URI of a strain passport can be viewed as a globally unique identifier for the strain number it represents. This identifier has the added benefit of being resolvable to a single page, which can be used to download additional information, either automatically or manually.

URI design

URIs for passports on the site can be decomposed in three main parts, some of them optional. Other projects such as the SeqRank project (described in Chapter 6) are afforded their own namespace which does not necessarily follow these rules. Table 2.2 lists the various entities that can have a passport on the StrainInfo website, along with an example URI illustrating how the address of each entity is constructed. All URIs for passports and browsers are built as follows:

```
http://www.straininfo.net/<entity>[/<id>[/<method>]]
```

Entity The type of object, e.g. strains, taxa, sequences, publications, genomes, ... Within the URI, the entity type is always written with a plural name.

Id The identifier of the entity. This can be different for different entity types.

Entity type	Id type	Example	URI
strain	culture ID	4490	`http://www.straininfo.net/strains/4490`
taxon	taxon ID	339	`http://www.straininfo.net/taxa/339`
sequence	accession number	AJ310096	`http://www.straininfo.net/sequences/AJ310096`
publication	publication ID	6005	`http://www.straininfo.net/publications/6005`
genome	bioproject ID	384	`http://www.straininfo.net/genomes/384`

Table 2.2: List of entity types and their identifiers.

Method Invokes a separate view or function. The most important method provided is the 'browser' method which opens the strain, taxon, sequence, publication or genome browser.

The search convention

Some of the entities are generally described elsewhere through well-known identifiers, that are not necessarily the same as the StrainInfo identifier, for example cultures for which the external identifier is a strain number and the StrainInfo identifier is a culture ID. In order to make it easy to link to such entities, each namespace has its own search function, which is accessed by replacing the ID in the URI defined above by the string 'search' with additional options. This search can support several search-specific query parameters (i.e. `/strains/search?strainNumber=LMG6923`) and one general parameter: target, which can be used to control the landing page of the search, e.g. if a partner database wishes to link directly to the browser, he or she can use target=browser. However, the default is always the passport. Smart search is invoked by accessing the root level `/search` and will search *all* entities.

Output Formats

The type of output format is controlled by fetching an URL and appending an extension. The only currently supported extra extension is the `.xml` extension which will return a strain passport in MCL [14] format. When no extension is given, the default returned format is XHTML.

2.5.3 Custom exports for bulk data exchange

Often, our partners are interested in a limited set of information for each strain that is distributed across the entire data set (all strains or all type strains). In this case, it is often not feasible to enumerate all pages in StrainInfo, which is also practically a heavy load on the StrainInfo servers. One of the main partners of StrainInfo is SILVA, for instance, which uses StrainInfo data to determine which sequences belong to a type strain. For such uses, custom data exports have been implemented that are made available on a separate page of StrainInfo, the exports page found at `http://www.straininfo.net/exports`, for any interested party to download and use.

3

The StrainInfo Data Integration Pipeline

3.1 Introduction

StrainInfo, as outlined in Chapter 2, links publicly available strain subcultures to a particular strain, and with them publications, molecular sequences, taxonomic information and genome projects. The principal service provided is not the content of those individual data records, but the mapping between them. This type of service was described in [38] in terms of a knuckles-and-nodes architecture. The nodes represent source databases, in this case BRC culture catalogues, the INSDC source databases, and other related databases. StrainInfo itself is a *knuckle*: a relationship service which indexes the other databases and provides users with information about the relationship between records in these various databases, in this case in the domain of cultured microorganisms.

The knuckles-and-nodes approach to mapping data can be contrasted conceptually with a data warehousing approach, where all data of possibly disparate databases is imported directly in a central database for integration and analysis. While StrainInfo does not necessarily need access to the full under-

lying data records in this way, most of the relevant services do not expose universal identifiers that can easily be cross-referenced. Data integration instead must evaluate the data records, sometimes repeatedly in response to likely errors in the source databases, using a previously published integration algorithm [10]. At the very least, this requires a subset of the source data to be cached by the data management system. In practice, most of the data is needed in order to provide a low-latency, quality-controlled and usable web interface to navigate the StrainInfo mappings, and so there is little practical difference between both approaches when it comes to basic data integration.

Since it is necessary for StrainInfo to maintain its mapping and website to import all data locally, just as in a data warehousing approach, it encounters similar challenges. Namely, data updates must be performed frequently, in the face of infrequent yet significant changes in every source database [39]. Single source databases tend to change their Application Programmer Interface (API) or data formats infrequently, but when a service relies on several such sources, data import breaking changes may occur monthly, weekly or even daily, depending on the number of sources integrated. Data must also be kept as up-to-date as possible, since source databases might evolve quickly, adding, modifying and deleting records. Users expect this data to be available in cross-linking services, so that they do not have to rely on their own searches to supplement the provided data records. Therefore, the updating mechanism must run frequently, precluding manual operation. It should also be able to notify curators or developers of changes in the data sources data format or schema, so they can react to changes in source databases. This chapter describes the construction of several data integration pipelines that have served to largely automate StrainInfo's integration of external data sources. It focuses mainly on the integration of sequence data from the INSDC database and taxonomic information sources, which was undertaken as part of this work. Integration of BRC catalogue information is undertaken by import of BRC supplied Microbiological Common Language (MCL) [14] data files, and has been described previously in [12].

3.2 Automation components

As mentioned in Chapter 2, StrainInfo as a whole is built on a variety of different technologies and thus so is much of the software that is used to integrate

various pieces of source information. Over the years, bits of functionality have been implemented in PL/SQL stored procedures, bash shell scripts, python scripts, Java language Maven modules, etc. Migrating all the legacy code to a single technology or programming language is too time-consuming to be feasible, nor applicable for all external software packages that are used in various places. Therefore, a solution is necessary that is able to deal with all of these different technologies with some intelligence, or at least allows access to a full shell environment in order to call the necessary programs or scripts, when no other option is available. It must be able to run pipeline tasks in a certain order, and there should be support for automatic triggering of this entire pipeline at set times.

All pipeline tasks also share some common requirements. It is firstly necessary to be able to maintain underlying source code for specific tasks easily. Source code for all components is normally maintained in version control software, in this case a Git[1] or Subversion[2] repository. A common framework can take care of tedious tasks, such as the periodic checking for program updates in these various different version control systems, as well as handling the details of downloading and applying those updates when they become available. Much of StrainInfo's code is built with the Maven build tool, integration with which provides the ability to manage software dependencies without manually installing them, and the guarantee that the environment in which code on the server runs is the same as on a developer's machine. Direct integration of Maven functionality in the build tool makes this possible without requiring reproducing Maven settings in all scripts that handle Maven modules. Finally, many tasks means many opportunities for small day-to-day problems that might make a task fail. Whenever a failure occurs, there needs to be a mechanism that notifies the StrainInfo team about the failure, and functionality to find out what exactly happened, i.e. full logs of all program output from the failed task. In the StrainInfo pipeline, the Hudson Continuous Integration tool[3] has been selected to fill this niche.

Hudson Continuous Integration tool

Martin Fowler defines Continuous Integration as follows [40].

[1] Stored in the Ghent University GitHub instance available at `http://github.ugent.be/`
[2] `http://subversion.apache.org/`
[3] Available from `http://hudson-ci.org/`

> "Continuous Integration is a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily — leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible."

While this is a development practice, it does require tool support in order to run each automated build, and its tests, so that errors can be detected. Because of possible configuration differences between developer machines, it is generally considered best to run these builds in a separate location, usually on a separate server. In order for this to be useful to a development team, this Continuous Integration (CI) tool has to support a variety of different technologies, be aware of version control software, be able to automatically schedule builds and include notification support. There is also the need to keep track of what went wrong when a build fails, by capturing all process output. All these requirements overlap with those for running the pipeline. Therefore the pipeline was implemented as separate tasks inside a particular CI tool, specifically Hudson CI.

3.3 Building the StrainInfo and GRS software platform

In addition to running the data integration pipeline, Hudson is also used more prosaically within StrainInfo development in its original intended capacity as a Continuous Integration tool. Since all StrainInfo modules discussed in Chapter 2, and all Genomic Rosetta Stone (GRS) software, further discussed in Chapter 4, are implemented as modules defined by the Maven build tool, a definition of all their respective dependencies is available, they can easily be compiled and tested automatically when needed. Hudson monitors the source code repositories for all relevant modules developed as part of the StrainInfo and GRS projects, and builds new versions whenever changes become available. In addition to providing notification to developers when a compilation problem occurs or a test fails, this also makes it possible to automatically publish the compiled software artifacts to a separate repository, obviating the need for developers to download, compile and install such dependencies themselves, for instance in the case of the StrainInfo core module, which is used

by several different project. The software repository itself is managed using Sonatype Nexus[4], a server application which is aware of Maven configuration and repository layout conventions, bundling Java ARchive (JAR) files with so-called Project Object Model (POM) files, which are the Maven configuration files describing contents, version, dependencies and other metadata of said JAR file.

3.4 Data providers

As should be clear from Chapter 2, StrainInfo data is gathered from a variety of online sources. The main sources for data to be integrated are:

Biological Resource Centers (BRC) Data from BRCs was in the past gathered automatically via screen scraping. However, due to the high maintenance cost, this is now done in the form of a push architecture, where individual BRCs upload XML information to StrainInfo, which is then integrated into the data set. This data is then subjected to XML validation checks, which encode many restrictions on the data content in order to enforce data quality, followed by a manual check for common errors by StrainInfo curators. The rate at which these updates occur varies with each individual BRC, as the uploads happen on a completely voluntary basis. Some BRCs update data bimonthly, others only yearly. As mentioned, the manner in which these files are gathered and curated is outside of the scope of this chapter.

European Nucleotide Archive (ENA) The International Nucleotide Sequence Database Collaboration (INSDC) [6] is composed of three member databases: GenBank [27], the European Nucleotide Archive (ENA) [41], and the DNA Databank of Japan (DDBJ) [29]. All three databases synchronise information, and so sequence data is in principle identical between all three. In practice, sequence record annotations sometimes differ slightly (as will become apparent in Chapter 5), likely due to minor differences in policy between databases. Regardless, StrainInfo imports the "PRO" (for prokaryotes) and "FUN" (for fungi) section of every full ENA 'release' (released roughly every six months) in order to be able to link molecular sequences to strains. In addition, ENA releases files containing new

[4] Available from `http://www.sonatype.org/nexus/`

and updated sequence information daily. This data is downloaded and integrated into StrainInfo with the same frequency. Both types of data export are downloaded through the `siseqparser` program, discussed in Section 3.5.3.

Taxonomic databases While StrainInfo bases type strain information on the list of Prokaryotic Names with Standing in Nomenclature (LPSN) [11] (currently available from `http://www.bacterio.net/`) and Mycobank [21], taxonomic names from four different taxonomies are supported: the aforementioned LPSN and Mycobank, the National Center for Biotechnology Information (NCBI) taxonomy [22] and the Prokaryotic Nomenclature Up-To Date[5], provided by the German Collection of Microorganisms and Cell Cultures (DSMZ). LPSN does not provide any structured data export files, and so its website is imported into the database using a so-called 'screen scraper': a program which automatically traverses links and interprets HTML pages intended for human, rather than automated consumption. Mycobank is not, at this point, updated through a Hudson CI task but through a manually launched in-database update task. The other two taxonomy resources do provide (easily) machine-readable data dumps, which are imported into the databases using custom scripts. All prokaryotic taxonomic source data is renewed weekly.

BioProject and NCBI LinkOut Projects to sequence the full genome of a single organism can be registered as a Bioproject at the NCBI database [32]. Important metadata is available as part of these projects, and they can be used as a starting point to locate more. A data export of the BioProject database in eXtended Markup Language (XML) format is made available data from the NCBI FTP servers. This data is integrated weekly by the `gpdler` and `grsloader` applications. It is combined with mapping information to external databases downloaded from the NCBI LinkOut system. These mappings are used by the Genomic Rosetta Stone to build a mapping service of individual service identifiers.

[5]The Prokaryotic Nomenclature Up-to-date can be found at `http://www.dsmz.de/bacterial-diversity/prokaryotic-nomenclature-up-to-date.html`

Task group	Update frequency
Building StrainInfo libraries	on demand
MCL imports	on demand
Sequence record imports (siseqparser, ferede)	daily
Taxonomic database imports	weekly
GRS updates (gpdler, grsloader)	weekly
Exports and sitemap	weekly

Table 3.1: List of the different groups of tasks run in StrainInfo and the frequency with which they are run.

3.5 Pipeline components

A variety of data import tasks handle the actual integration (and subsequent export) of information into StrainInfo. As shown in Figure 3.1, many of these tasks do not require that other tasks are run in advance. In practice, there are thus several parts of the pipeline that can be run concurrently. Import of sequence records from ENA (by `siseqparser`), and the detection of molecular sequences in those records which were submitted as their reverse complement (by `ferede`), is run daily, while taxonomic updates happen weekly. Linking publications and sequences to strains is done as part of the `siseqparser` import. However,further linking of cultures to sequence records can be done when information from LPSN, DSMZ, and NCBI taxonomic databases has become available. These extra links are established by the "link cultures" tasks, after which the "update publications" task downloads publication information from the NCBI PubMed publication database. Both are run weekly after completion of the taxonomy tasks.

Several of these integration tasks trigger the construction of export files for consumption by external users, for instance the NCBI LinkOut system, which provides links to StrainInfo passport pages on NCBI sequence records. The executions of export tasks is only triggered when necessary, i.e. when underlying data has changed. The various tasks are described in more detail below. All these different tasks can be roughly divided into only a few groups, which are run as one whole or in quick succession. Table 3.1 summarizes the types of data import or software building tasks, and the frequency in which they are run.

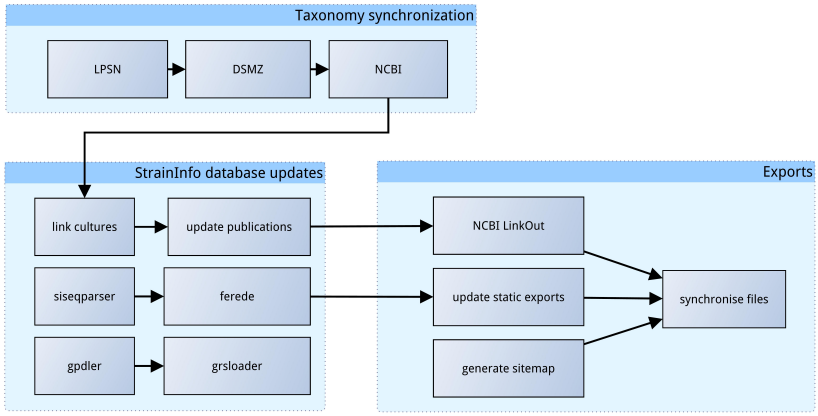


Figure 3.1: Subtasks involved in StrainInfo related imports, data integration and exports. Two tasks are connected if completion of one triggers the execution of the other. While `gpdler` and `grsloader` are two tasks linked to the Genomic Rosetta Stone system, they are shown here since the data they gather into the database can also be re-used for StrainInfo genome passports.

3.5.1 LPSN

LPSN is a strictly curated and complete source of taxonomic information for Bacteria and Archaea, listing detailed strain and recommended 16S rRNA gene sequence information with each taxonomic name in the index. Unfortunately, no machine readable data export exists for the taxonomic information contained in the pages of LPSN. Therefore, the lists of all taxa at or above the rank of genus are used as input to an automated program, commonly called a spider, that traverses links to taxon pages and reformats data in a structured format for loading into a database server. The crawler was implemented as a set of parsing rules and logic in the scraping framework Scrapy[6].

Scrapy includes the `lxml` parsing library, which exposes HTML code as a tree-like structure that can be queried using Xpath, a declarative query language for parsed XML data. Unfortunately the code of pages on LPSN was never meant to be parsed by machines. As a result, the spider implementation augments the use of such queries with various heuristics based on font, text colour, and other clues contained in the text. In this manner, the spider can obtain a

[6] Available from `http://scrapy.org/`

list of taxonomic descriptions, each including a taxon name, taxon rank (e.g. whether it is a species, genus, or higher), list of type strain numbers, 16S rRNA gene sequence accession number, various references and etymology data. In the practice of taxonomy, species names are often moved to different genera, merged with other species and so on. This synonymy, as well as the identity of a possible neotype strain, is also parsed and stored.

After parsing, each parsed item, corresponding to a single taxonomic name, is run through a pipeline which removes faulty records and fields using a set of filtering rules. After this step, all items are output to a simple Comma Separated Values (CSV) file, from which they are uploaded into the database using the Oracle `sqlldr` utility. The final step is the linking of strain numbers found in these records with the cultures as stored in the StrainInfo database, done by parsing strain numbers and generating identifiers. This is performed by a dedicated PL/SQL stored procedure, previously available in the database.

3.5.2 NCBI and DSMZ taxonomies

Taxonomic information is linked by depositors to INSDC sequence records. The NCBI taxonomy is a curated data set of all these taxonomic names [33] and provide an entry point to users looking for genetic information linked to a particular taxon. Its names and identifiers are integrated into the StrainInfo database by a legacy script which downloads a full data dump of the taxonomy, freely available from the NCBI FTP servers[7], and stores names and identifiers in the database in order to be able to link to the NCBI taxonomy browser from any taxon passport in StrainInfo. Only known prokaryotic or fungal names are made available in StrainInfo.

The DSMZ provides a taxonomy called the "Prokaryotic Nomenclature Up-to-date". It is another curated taxonomy based on the publication record of Bacteria and Archaea, published monthly in a Microsoft Excel file containing all current entries. The taxonomy includes similar information as LPSN, although these taxonomies sometimes differ on certain names (most frequently, it appears, in the spelling of new species names). In order to give users access to different taxonomic opinions, it is integrated into the database with a simple shell script and database procedure, in similar manner as the NCBI taxonomy, and appears on any taxon passport for a recognized name in StrainInfo.

[7] At time of writing, the public FTP servers are found at `ftp://ftp.ncbi.nih.gov/pub/`

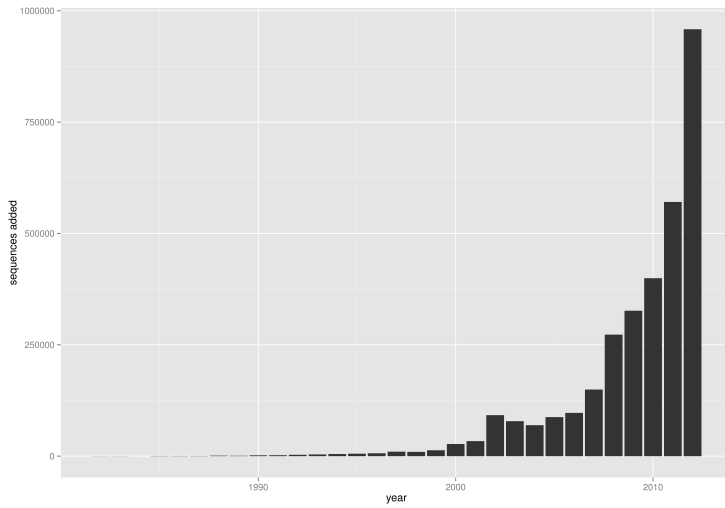


Figure 3.2: Sequences from the prokaryotic and fungal sections added into the European Nucleotide Archive by year, from its first 98 sequence records in 1982 through to the year 2012. Dates were based on the sequence record creation date.

3.5.3 ENA sequence records

Almost all publicly available genetic information is available as (sets of) individual sequence records from the INSDC member databases. Much of this molecular sequence information is indispensable for modern microbiology, as is explored further in Chapter 5. As part of their mission, all INSDC member databases provide publicly available structured data files of the entire database. The total size of this archive is now measured in petabytes and growing exponentially, doubling roughly every eight months [28, 42]. StrainInfo, as it focuses on microbial information, only imports a small subsection of all this data, namely the prokaryotic and fungal sections of the databases and associated whole genome shotgun sequencing data. This data has section identifiers "PRO" and "FUN", which at time of writing still adds up to 38GB of gzip compressed data, a number that is steadily increasing as should be clear from Figure 3.2.

The `siseqparser` sequence record importer was built with two major goals. The first, to provide a robust parser for all information contained in an ENA sequence record file. The format of these files is sometimes referred to as EMBL

or EBI sequence format, while in the current nomenclature used by ENA it is simply referred to as the ENA sequence format. Software module names predate this change in nomenclature and still refer to it by the EMBL name. The chosen programming language for the parser is the Python programming language[8]. It was developed independently from scratch since the only other robust Python language parser for these files found at the time was that of the Biopython project[9], which lacked full support for the metadata available in ENA format files. Parsing functionality is provided in practice by the `embl_parser` module.

The second goal of `siseqparser` is to leverage the `embl_parser` module to provide a full pipeline for integration of EMBL information with that found in the StrainInfo database. This requires tracking functionality for the download of new ENA data files as they become available from the EBI FTP server, logic to decompress and parse files, outputting the results into a set of tab separated files which can then be uploaded into the StrainInfo database, and support to access the existing legacy functions residing in-database for data integration. In practice, a Python script called `si_embl_updater` handles the details of these actions.

embl_parser

Listing 3.1 is a typical example of a sequence record in the ENA format. Note in particular the `ID` line, which contains the accession number used to link to sequences on the StrainInfo website. An ENA record also contains creation and modification dates, which are used to conditionally update sequence records in the database. Feature descriptions (lines starting with `FT`) identify the source strain and the genes found in the sequence record. The feature annotations are instrumental in linking the sequence record to a culture, and identifying genes for applications in StrainInfo that need to retrieve them. In this case, the chosen sequence record contains a sequence of a section of the 16S rRNA gene.

An ENA format file is highly structured and designed to be parsed line by line in a single pass of a parser, with an enforced maximum line length for parsers which rely on fixed length buffers. This makes the implementation of `embl_parser` quite straightforward. It proceeds through the file line by line, merging lines when data has been split over several lines, and assigns the values

[8]Available from `http://python.org/`
[9]Available from `http://biopython.org/`

```
ID   AJ320493; SV 1; linear; genomic DNA; STD; PRO; 1521 BP.
XX
AC   AJ320493;
XX
DT   30-AUG-2001 (Rel. 68, Created)
DT   13-SEP-2001 (Rel. 69, Last updated, Version 3)
XX
DE   Paenibacillus polymyxa partial 16S rRNA gene, strain DSM 36T
XX
KW   16S ribosomal RNA; 16S rRNA gene.
XX
OS   Paenibacillus polymyxa
OC   Bacteria; Firmicutes; Bacilli; Bacillales; Paenibacillaceae; Paenibacillus.
XX
RN   [1]
RP   1-1521
RA   Sproeer C.;
RT   ;
RL   Submitted (28-AUG-2001) to the INSDC.
RL   Sproeer C., Molecular Systematics and Ecology, Dsmz, Mascheroder Weg 1B,
RL   38124 Braunschweig, GERMANY.
XX
RN   [2]
RA   Suominen I., Sproeer C., Kaempfer P., Lounatmaa K., Salkinoja-Salonen M.;
RT   "Paenibacillus stellifer sp. nov., a cyclodextrin producing species
RT   isolated from paperboard";
RL   Unpublished.
XX
DR   CABRI; DSM 36.
DR   EuropePMC; PMC129906; 12406710.
DR   RFAM; RF00177; SSU_rRNA_bacteria.
DR   RFAM; RF01959; SSU_rRNA_archaea.
DR   SILVA-SSU; AJ320493.
DR   StrainInfo; 39157; 1.
XX
FH   Key             Location/Qualifiers
FH
FT   source          1..1521
FT                   /organism="Paenibacillus polymyxa"
FT                   /strain="DSM 36"
FT                   /mol_type="genomic DNA"
FT                   /note="type strain"
FT                   /db_xref="taxon:1406"
FT   rRNA            <1..>1521
FT                   /gene="16S rRNA"
FT                   /product="16S ribosomal RNA"
XX
SQ   Sequence 1521 BP; 385 A; 351 C; 488 G; 297 T; 0 other;
     ggctcaggac gaacgctggc ggcgtgccta atacatgcaa gtcgagcggg gttagttaga
60
     agcttgcttc taattaacct agcggcggac gggtgagtaa cacgtaggca acctgcccac
120
```

Listing 3.1: First 50 lines of the sequence record with accession number AJ320493, in ENA text format.

found in those lines to a set of lightweight classes that associates a field name with each data value. The main information in each feature record is stored in the `EmblRecord` class, while features and publications listed in the record are parsed separately as lists of `EmblFeature` and `EmblPublication`, which are attached to the `EmblRecord` instance.

Entries are usually bundled together in large data files, separated by the string "`//`". Because of the amount of data involved in parsing even only one such file, `embl_parser` provides several functions to use it in streaming mode. These accept a file name or file stream and buffer lines until a record separator has been found. The record thus obtained is parsed and processed, before proceeding to the next record. By making use of the generator pattern, available in Python through the `yield` statement, a caller can iterate through the results returned by these functions as it would a normal list of records, while only ever keeping at most two records in memory. Listing 3.2 illustrates how this works in practice: clients iterate over the results of the `parse_file_stream` method which returns `EmblRecord` objects, returning control to the caller when a new record becomes available.

conditional update

The `si_embl_updater` script expands on the functionality supplied by the parser. Each time it is run, it checks whether a local cache of downloaded files is still valid by checking for missing files and using EBI supplied Cyclic Redundancy Check (CRC) files to check if downloaded files match those available on the server. New files are automatically downloaded. In addition, the EBI also provides update files, which follow a predictable naming pattern. When new files of this type become available for the downloaded release, they are also downloaded. Any files that have since been removed (i.e. after a new release) are also removed from the local cache.

Not all records so obtained may have been updated (for instance, a new release will also contain all older sequence records). Therefore, a list is gathered of all accession numbers currently in the database, along with the last modification date of their current sequence record. This list is used in the next step to decide whether to save a record for update into the database, or simply skip it.

Since they are offered in compressed format, the newly downloaded files from the first step are passed, one by one, to the `gzip` program for decompres-

```
def _extract_records(filestream):
    """
    Extract EMBL records from a file stream.

    This method returns the records present in the supplied embl-format file
    as list of strings comprising each record, to be parsed further by other
    functions.
    """
    line = filestream.readline()
    while line != "":
        cur_record = []
        while line.rstrip() != "//":
            cur_record.append(line.rstrip())
            line = filestream.readline()
        yield cur_record
        line = filestream.readline()

def parse_file_stream(stream):
    for record_source in _extract_records_stream(stream):
        yield SeqParser(record_source).parse()
```

Listing 3.2: Use of the generator pattern to reduce memory usage when extracting records from a file. Client code can call the `parse_file_stream` and iterate over the results directly, without loading each item into memory.

sion. The file stream returned by `gzip` is then passed to `embl_parser` which parses these into records. Each record's accession number and modification date are checked against the previously retrieved list of accession numbers and their modification. A reformatted version of the record is written to disk for local update when changed.

Finally, Oracle SQL loader is called to upload the records and associated features and publications into the database. A PL/SQL stored procedure is called to identify strain numbers listed in the records and use those to link records to cultures in StrainInfo.

When update files are involved, different versions of sequence records might occur more than once in the input files, e.g. when update files are parsed together with the release files, or when the sequence record has been updated several times since the last release. This is handled by first scanning through the files and keeping track of accession numbers that appear more than once. The script can then simply skip previous occurrences in the record, to avoid

writing multiple entries out to disk, which then would have to be deduplicated elsewhere.

3.5.4 Feature reversal detection (ferede)

Molecular sequence features deposited into the INSDC databases should be deposited in the 5′ to 3′ direction (after translation via the feature annotations), as is the convention when denoting a single strand. However, there is a small contingent of sequences which were indeed mistakenly submitted as their reverse complement, i.e. in the opposite direction and with purines and pyrimidines transposed. This is especially a problem when working with 16S rRNA gene sequences derived from sequence records in applications such as StrainInfo SeqRank, which includes a similarity criterion (see Chapter 5), since performing a multiple alignment between sequences with different orientations will naturally result in nonsensical results.

Determining the sequence in the correct direction is a straightforward operation (requiring simply reversing the reverse complement and transposing purines with pyrimidines and vice versa), but it requires one to know the sequence is reversed first. Therefore, 16S rRNA gene sequences derived from newly imported sequence records by `siseqparser` are scanned with V-REVCOMP [43], which attempts to find 18 different domains in the rRNA gene (detected through matching of a set of profile hidden markov models with HMMER [44]), in order to detect if a sequence is likely reversed. Ferede (for FEature REversal DEtection) manages the fetching of sequences, execution of V-REVCOMP and interpretation of results. The sequences assigned the status “reverse” or “uncertain-reverse” by V-REVCOMP are not physically turned (as they are only one of the features of what could be a longer sequence) but marked as being the reverse complement in the database. When 16S rRNA gene features are retrieved in the StrainInfo core module, this information can be used to reverse complement sequences before alignment is undertaken.

3.5.5 StrainInfo exports

StrainInfo offers a number of static database exports, most available on the exports page (available from `http://www.straininfo.net/exports`). These exports are generally simple database dumps with relevant information such as marking whether a molecular sequence is type strain related (used by the SILVA

team in the construction of their database releases). In addition, StrainInfo provides a set of site map files, used by search engines to find and index the many available passport pages and also uploads a set of custom files to the NCBI LinkOut system and the ENA repository, which they use to link sequence records to relevant StrainInfo passport pages for the source material. On-the-fly creation of all of these, often very large, files would introduce a high database load when multiple requests occur. Moreover, it is unnecessary for these files to be updated more than integration occurs, and thus they can be safely created daily or weekly, depending on the type of data and application. Therefore, most tasks that alter data in the database trigger creation of new exports and upload to the relevant locations (the StrainInfo websites or the data servers of external partners). When updates should occur even more infrequently, they are scheduled separately on a biweekly basis.

3.5.6 GRS

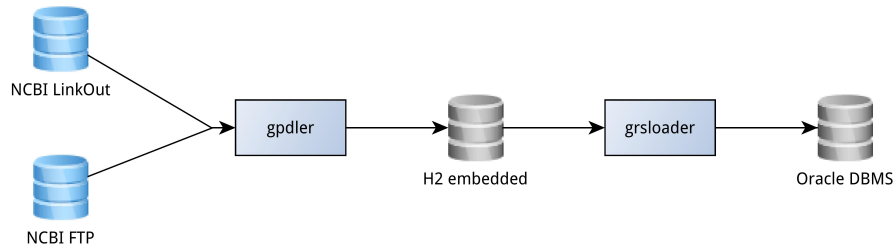


Figure 3.3: Data flow from NCBI FTP servers (serving the BioProject XML export) and NCBI LinkOut through the `gpdler` application to a local H2 embedded database file. This file is then loaded by the `grsloader` application, which uses it to transfer the data to equivalent data tables in StrainInfo's Oracle DBMS.

Genome project source organism related information is published by the NCBI as a set of BioProjects. A complete XML format export of this database is available from the NCBI FTP servers and can be parsed by the `gpdler` application, and additional mappings to other resources can be downloaded from the NCBI LinkOut system. `gpdler` stores this data in a relational database, where it can be used by StrainInfo and the Genomic Rosetta Stone system. Since both

the Genomic Rosetta Stone web application and `gpdler` are open source, it was elected to store the data in a freely available embedded database, the H2 database engine[10]. However, StrainInfo makes use of the Oracle RDBMS, and can also make use of the same information. Therefore, the data is inserted into the Oracle database by the `grsloader`, which utilises `gpdler` as a library containing both an object relational mapping for the database, and access methods for the embedded database it creates.

[10] Available from `http://www.h2database.com/`

4

The Genomic Rosetta Stone

4.1 Introduction

A rapid fall in sequencing costs has made it feasible for small- and large-scale studies alike to increasingly rely on fully sequenced genomes for their analyses. As a result, the number of published genomes is rapidly increasing [6]. Additionally, researchers continue to devise new ways to examine and to analyze genomic data. Increasingly, genomic data analyses rely on additional related data about the organism, the original sampling event and the subsequent cultivation procedures. Recently, the Genomic Standards Consortium published the Minimal Information about any (x) Sequence Standard (MIxS) [45]. However, often the additional related data is not available with the original sequence record. Typically either because the information was not available to the original submitters, or because the metadata is curated and maintained by separate and specialized database providers. Various data providers already provide this data freely through web sites or Application Programming Interfaces (APIs). Such resources now abound, and are an important part of the genomic research ecosystem.

The availability of related data distributed over heterogeneous data providers has naturally led to a desire among researchers and software developers alike

to build new services that allow them to download and browse all available related data on genomes. Furthermore, there is a growing interest in automated analysis of large data sets, where the data quantities involved preclude manual downloading and re-formatting of genome-related information. However, genome-related data is scattered across various data providers, operating within the constraints of different problem domains, be it genomes themselves, annotations, publications or materials in culture collections. Each problem domain comes with a different way of identifying resources, and of linking them to genomes. Additionally, their increasing number means data providers do not, and should not have to, know about or link to each other. Therefore, integrating information found on one resource with records on another resource still requires laboriously navigating between isolated data islands.

The distributed nature of data providers means any solution to the cross-referencing problem must be updatable and maintainable. Likewise, it should be searchable by allowing the use of any genome or genome-related data identifier when searching for related information. The Genomic Rosetta Stone (GRS) [46] achieves this goal in a dual way: it standardizes the use of the BioProject identifier [32] as a single identifier — and provides a mechanism and tools for fetching related records based upon the use of the LinkOut system operated by the National Center for Biotechnology Information (NCBI) [33]. The goal of the GRS is to create a stable method for lookup of additional genome-related data in whichever format necessary and make it freely available as open source software.

This chapter discusses the architecture of the current system, as well as the registration process for new partners and the current data providers participating at launch. The second part of this chapter examines several applications that currently make use of the GRS data, including StrainInfo, as well as a simple mashup constructed to demonstrate the capabilities of the new web API. It is shown that it is now quite straightforward to find related data. However, challenges do remain in the interpretation and integration of that data. The Genomic Rosetta Stone in its current operation constitutes a building block for the development of applications that explore and exploit genomic information, and may point the way to further improvements to existing data exchange practices.

4.2 Architecture

The Genomic Rosetta Stone links records from genomics-related databases to each other by mapping the identifiers of those records to each other. In order to establish this mapping, the GRS has chosen to map all database-specific identifiers to a common identifier: the BioProject identifier [32, 46]. From that mapping, all other mappings can be derived, by considering the set of all objects that refer to the common identifier as a single cloud of genomic information.

An advantage of BioProject identifiers is that a stable and freely accessible resource already exists to map them to outside data sources, namely the NCBI LinkOut system. There, data providers can update their own mappings, using the LinkOut FTP upload server and a specialized data format. Any data provider that wishes to join the system should register with NCBI LinkOut and upload a description of its mappings, possibly using existing mappings already in the system. They are given a provider identifier and can submit their own name and name abbreviation into the system. This abbreviation is later used in the GRS Resolver. Mappings are uploaded in a specialized XML format documented by NCBI, consisting of object ID (or query) and URL pairs. Resource links from data providers are listed on BioProject pages as part of the normal operation of the LinkOut system. They are also available through an open XML-based interface, ensuring that data is freely available to all interested partners, and that new services, such as the GRS Resolver, can be built on top of it.

As illustrated in Figure 4.1, advanced users can query the NCBI LinkOut system directly using the Entrez Programming Utilities (E-utilities) [33], which allow for the querying of mappings using a BioProject identifier. They do not provide any interface to query for BioProject identifiers when only a provider record is known, however, nor do they allow the querying of the resources associated with an identifier. To allow querying in both directions, the GRS Resolver (https://github.com/wdesmet/grs-web) was developed. It provides an easy-to-use, Representational State Transfer based (RESTful) [37] interface for the querying of mappings, and is supported by a mapping downloader, 'gpdler' (`https://github.com/wdesmet/gpdler`), which synchronizes BioProject information and mappings when they become available in LinkOut, using E-utilities. The Resolver and its synchroniser are open source and can be downloaded and run locally, or an up-to-date copy can be accessed online using any publicly available endpoint, such as the one run under the auspices

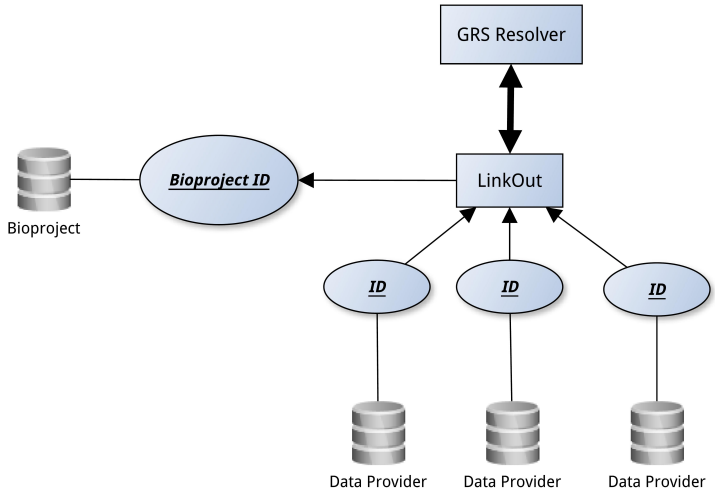


Figure 4.1: The Genomic Rosetta Stone maps identifiers to each other using the BioProject identifier as a common identifier. These mappings are stored in LinkOut, which can be queried for mappings from BioProject identifiers to data provider records. The GRS Resolver indexes LinkOut mappings and extends its functionality by providing reverse mappings.

of the StrainInfo project (`http://grs.straininfo.net`).

4.3 Data Providers

Table 4.1 lists all collaborators participating in the GRS at the time of writing. Their databases describe the genome projects themselves, properties of (partial) genome sequences or extend more widely to other genome-related information, such as, in the case of StrainInfo, the cultured microbial strains genomes were derived from. Each of these services might well refer to each other, and through GRS this kind of cross-referencing can easily be done on the fly if providers so wish.

Provider	URL
StrainInfo	http://www.straininfo.net/
GOLD	http://www.genomesonline.org/
RDP	http://rdp.cme.msu.edu/
SILVA	http://www.arb-silva.de/
GeminaDB	http://gemina.igs.umaryland.edu
IMG	http://img.jgi.doe.gov

Table 4.1: List of current GRS data providers. Additional providers currently listed on http://grs.straininfo.net/providers: jcrventer, dryaddb, vectorbase ; in preparation: HMP DACC

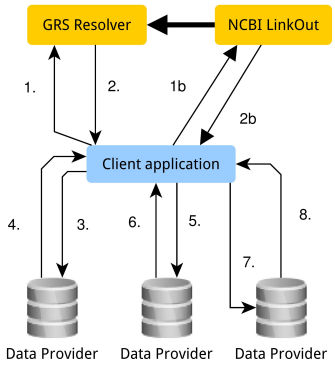


Figure 4.2: Example request and response using the GRS Resolver. A client application queries the Resolver application for a list of identifiers (1–2), using a data provider ID of any participating data provider. It then continues to query data providers separately (4–8), using identifiers returned in step 2. A client may also decide to query LinkOut directly (1b–2b), if it has access to a BioProject identifier.

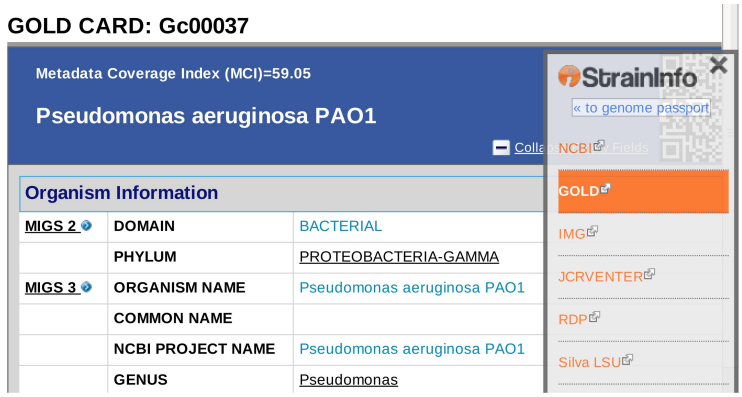


Figure 4.3: screenshot of StrainInfo genome browser.

4.4 Client Applications

Figure 4.2 outlines how a client application might make use of the GRS APIs to discover the location of genome-related data provider records. Of these, access to NCBI LinkOut through Entrez Programming Utilities (E-utilities) using the venerable 'elink' utility allows fetching and displaying the same external links as shown on an NCBI BioProject page. It is the oldest, and thus the most used in current applications, such as the browsers developed by both the Ribosomal Database Project (RDP) [47] and StrainInfo ([48] and Chapter 2). In the case of RDP, this browser lists genomic information and links to outside information sources, using LinkOut information. StrainInfo uses a similar mechanism, where each genome has its so-called passport page, and the genome browser, activated by clicking an external link, displays links to all resources in a browser toolbar, shown on the right in Figure 3, allowing quick navigation and comparison between the resources provided by various data providers.

4.4.1 Resolver API

New applications can be built using the GRS Resolver RESTful interface, which provides access to representation of remote resources — in this case identifier mappings and related information — through use of the HTTP protocol and simple URIs organised into namespaces. The major access points of the API are summarized in Table 4.2. The first of these is the 'projects' namespace, which

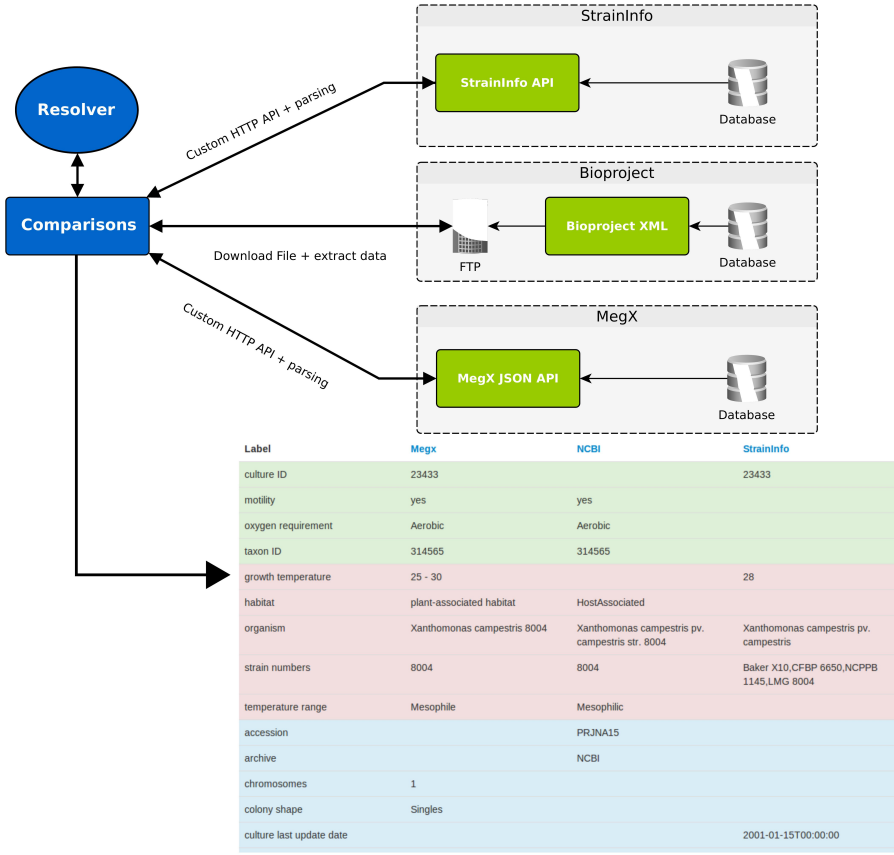
Namespace URI	Type of identifier	Example URI	Result
/projects	Bioproject Identifier	/projects/3	List of mappings
/mappings	Provider abbreviation	/mappings/straininfo	List of mappings to StrainInfo resources (full list)
/mappings/<provider>	Provider Identifier	/mappings/gold/gc00080	Mapping information and all related mappings
/providers	NA	/providers	List of providers with relevant attributes.

Table 4.2: List of namespaces and query patterns, which can be used to query the GRS web application at `http://grs.straininfo.net/`.

organises Bioproject information managed by the GRS application. In the current version, this includes only the list of all mappings (URLs to data providers) for a requested project. It can also be used at the top level to retrieve a list of all project identifiers currently recorded. Likewise, the 'mappings' namespace can be used to retrieve a list of all mappings, but is also organised in sections, each section containing mappings for a certain provider. The representation of a single mapping also contains all related mappings, facilitating quick retrieval of all identifiers when only the identifier of only one provider is known. The 'providers' namespace, finally, can be used to query for provider information, including the provider name abbreviation, which as a parameter in querying the mappings namespace.

4.4.2 Case Study: A Simple Comparison App

Since many data providers provide resources in similar domains, it is natural to wonder in what way their data overlaps, complements or even conflicts. Therefore, the Resolver API above was used to implement a simple comparison mashup as a case study of how the GRS API might be used in mashup applications. It displays information related to a genome project in a grid, using data from three different providers, namely NCBI's source BioProject data, Megx.net, a portal for integrated environmental and (meta-)genomic data intended for use in marine microbial ecology [49], and StrainInfo, an integrated database of cultured microbial strains. The grid resulting from querying and mashing up the data of these three providers, shown in Figure 4.4, lists in its rows different terms found in the data of these providers, and in its columns values for those terms found at each respective data provider's API or data



Label	Megx	NCBI	StrainInfo
culture ID	23433		23433
motility	yes	yes	
oxygen requirement	Aerobic	Aerobic	
taxon ID	314565	314565	
growth temperature	25 - 30		28
habitat	plant-associated habitat	HostAssociated	
organism	Xanthomonas campestris 8004	Xanthomonas campestris pv. campestris str. 8004	Xanthomonas campestris pv. campestris
strain numbers	8004	8004	Baker X10,CFBP 6650,NCPPB 1145,LMG 8004
temperature range	Mesophile	Mesophilic	
accession		PRJNA15	
archive		NCBI	
chromosomes	1		
colony shape	Singles		
culture last update date			2001-01-15T00:00:00

Figure 4.4: Operation of the GRS Comparison app. Comparisons are generated by querying GRS data where necessary and extracting records from various APIs from the three supported data providers. The eventual result is shown in a grid system, for easy at-a-glance comparisons.

export. By colour coding the rows, it is easy to see at a glance which terms are only defined by one data provider (blue colour), on which terms at least two have data and are in agreement (green colour) and on which they seem to disagree (red colour). Data integration is done on-the-fly using known information from the included data providers. The mashup is available at `http://grs.straininfo.net/comparisons`, and its source code is available to developers as part of the GRS Resolver source code.

4.5 Discussion

4.5.1 A Common Identifier

Linking multiple disparate resources together, such as in GRS, might simply be done by keeping track of which records refer to each other, but that solution is quite storage intensive; in the worst case, where many records refer to each other, it will require storage space that grows quadratically with the number of data providers, even though all records are intrinsically linked to the same objects, namely genomes. Furthermore, integration of the data records in an application based on GRS would still require its developer to internally assign some sort of singular identifier to each genome. Therefore, it is more advantageous to choose a common identifier for each genome in advance. With this identifier in place, the only mapping required is that from each data provider's identifier scheme to a common identifier.

A good central identifier should be universally applicable to, and available for, all genomes. Ideally it would refer to every individual genome, from initial sequencing to assembly through to archival, and remain stable, i.e. it does not change over time or through the various stages of a project. No 'perfect' identifier exists for genomes. Rather than attempting to solve this problem by assigning yet another new identifier, the GRS has chosen to use the BioProject identifier. These identifiers exist for almost all genomes, are generally stable, never reused and map to a small number of genomes per identifier. Often data providers can easily establish a mapping to BioProject identifiers, and some already do. Data providers that cannot link their data to a BioProject identifier, can often link it to the data records of other data providers. By making use of the GRS, they can also easily retrieve a BioProject identifier to use in their mapping. By further leveraging NCBI Linkout, the onus of updating mappings falls on the data providers. This distributed update system will allow the GRS to grow organically, from the data providers out, with no further maintenance required on behalf of the GRS core working group.

4.5.2 GRS Resolver

The GRS has a dual goal: to make finding genomes and genome-related information much easier, and to facilitate the building of new applications that integrate this information. To attain those goals, developers require an Ap-

plication Programming Interface (API) with which they can easily build new applications for the visualisation and analysis tasks required by end users. Such an API must be comprehensive, concise and easy-to-use if it is to see significant adoption numbers. Preferably, it should also allow as many different programming styles and use cases as possible.

E-utilities are the tool of choice for developers already familiar with them, providing access to the same information as the GRS Resolver. They do not, however, perform any additional processing on mappings, and only allow developers to query for records using the BioProject identifier or a specialized query. The GRS Resolver enables additional functionality, by parsing URLs for data provider identifiers heuristically, and providing full identifier mapping from any resource record to another, as was shown in Figure 1. In addition, it provides data in three commonly used data formats, namely plain text, ad-hoc eXtensible Markup Language (XML) and JavaScript Object Notation (JSON). This design eschews interface complexity traditionally associated with other web service designs, such as the Simple Object Access Protocol (SOAP) [36]. Instead, queries use simple HTTP GET requests, resources are identified by their URI, and the required data format can be negotiated using request headers and an appropriate Internet media type. For instance, finding and processing a list of related mappings using a BioProject identifier or a mapping identifier, can be done using only information from Table 4.2 and standard HTTP functionality available in most programming languages, command line interface, or any web browser. This allows application developers to query the service in whatever fashion they feel comfortable with, maximizing convenience and lowering barriers to further development.

Any entity wishing to set up the GRS Resolver for internal use or as a service to the community can easily do so, since they are open source and freely available. This ensures that, should one endpoint become slow or suffer temporary downtime, users can easily switch their applications to a different endpoint. Furthermore, the synchronisation application downloads all relevant data about mappings into an embedded H2 SQL database, which can easily be integrated with existing relational database tools. This approach provides developers and users with a rich array of options with which to access the underlying mappings, allowing them to easily add new mapping formats or research other improvements to data provider interoperability.

4.5.3 Technical design

As the Genomic Rosetta Stone is intended as an open infrastructure, that could be used by many different, sometimes competing, types of users, it must be constructed with openness in mind. One of the main implications of this approach is that all software tools involved must be available and useful as separate modules in their own right, which reduces the reliance of data providers on potentially fragile hosting of other data providers. This approach is followed for all parts involved in the GRS system, except for NCBI LinkOut, which has a long and stable history as a central information repository. For the StrainInfo environment specifically, this precludes incorporating GRS directly into the StrainInfo web platform, and it equally follows that, if possible, both the downloader application and the GRS Resolver should not rely on the availability of an Oracle database, which only happens to be the data store of choice for StrainInfo itself. Therefore, database access is abstracted through the use of an object-relational mapping or ORM tool (provided by the Hibernate[1] framework), which hides any database-specific details from the implementation of all other classes. In addition, both include built-in support for the H2 Database Engine[2], which provides a database in absence of a user configured one.

In any hosting environment, using several different database types, as well as managing the access of various applications to different databases, induces a certain maintenance cost. At the same time, the H2 embedded SQL database, while convenient, is not meant for heavy server workloads. Therefore, data is not kept solely in the embedded database after downloading, but loaded into the central Oracle database. There are several ways to do this, the most obvious of which would be to modify the configuration of `gpdler` to connect directly to an Oracle database. This additionally involves supplying an extra SQL script to permit creation of database tables into the Oracle database, which is a task not left over to the Hibernate framework because of efficiency considerations. This approach is certainly one of the most straightforward, but has as a major drawback that none of the automated support for the embedded H2 database would ever be used within the StrainInfo team, aside from any automated tests bundled as part of the `gpdler` application. Experience teaches that features which are not exercised frequently will eventually break, leaving the end users (who likely don't use an Oracle database) with a broken piece

[1] `http://hibernate.org`
[2] `http://www.h2database.com/`

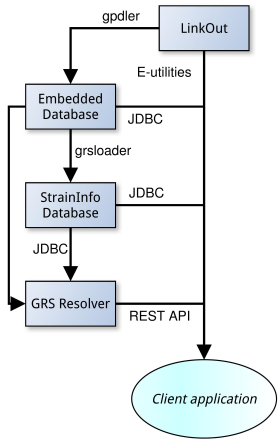


Figure 4.5: Diagram of the interactions of the various data repositories and software or APIs connecting them.

of software. Therefore, the perhaps less obvious decision was taken to always download the relevant records to an H2 embedded database first, making this the default supported situation, after which a separate application, internally named 'grsloader', is used to connect to this embedded database and transfer records to the StrainInfo central database in bulk. This exercises the default supported path supported to users, allows the GRS Resolver to be run using the more powerful Oracle database instance, and makes GRS data available to the StrainInfo platform, for use in the genome browser mentioned above.

Figure 4.5 provides a diagram of how the various pieces of software fit together with the databases defined. Each repository in this figure simply holds a transformed version of the data in the other repository. By transforming this data and providing additional functionality, however, that data can be made much more useful, since it relieves application writers of the need to write adaptation layers to provide similar functionality.

4.5.4 Mashup applications

Whereas single-source applications are generally concerned with providing a full-fledged toolbox to their users, mashups tend to excel at a single task, deriving value mainly from the way that they bring data from disparate sources

together. The mashup developed for this work is no different; it provides value by aggregating data from several data providers and analysing differences. Such a simple way of flagging problems and locating complementary information could not be built so easily without an application such as the GRS, which has made finding the data records much easier. The comparison so generated benefits users and curators alike, as it quickly becomes clear where data is missing, inaccurate, or inconsistent across databases, with minimal effort to launch the application.

There are still some significant challenges to the process of building a mashup in this domain. The first is the proliferation of APIs associated with any cross-domain service. As can be seen in Figure 4.4 for the example mashup, all three data sources offer their data in different ways. For the source data, it was elected to use a data export available from the NCBI's FTP servers, rather than directly querying E-utilities, since the data export exposes more information than is available through E-utilities. Megx.net, on the other hand, has a custom API which can be queried using a project identifier. In StrainInfo, culture data is identified by a culture identifier, which was found by using the GRS API defined above. Unfortunately there is no agreed-upon standard to expose such information through an API, even though each data provider shares information publicly through a web page or web portal.

Once data is downloaded from these APIs, the application logic builds a grid by comparing data for similar terms. While many of those terms are in the same domain, they do not share a common vocabulary. Any sort of data integration between services thus must develop its own mapping of terms, as was also done for the comparison application, which does this by being aware what each data source provides, and using hand-crafted mappings to know what should and what should not be in the same column. The mapping problem is a common issue with data exchange, where much of that data is described according to a locally defined schema and multiple data providers often have slightly different interpretations of what a term should mean, how it should be represented, or which values should be allowed. This situation increases implementation effort and precludes certain types of automated inference, unless further developer intervention is applied.

4.6 Conclusions

As the example of a simple mashup makes clear, the GRS and its Resolver can be used to quickly build new applications that integrate genomic data. However, while data providers already describe their data using various data standards, the data landscape remains extremely scattered, and there is no single standard way to retrieve a machine readable data record from any one provider. The Genomic Rosetta Stone can be seen as a basis for the development of further interoperability standards, particularly focusing on the availability and format of machine readable information. This would enable future applications to be implemented more rapidly, focusing on data analysis rather than data parsing. Crucial to such an effort is the development of a standard for web application access, preferably at the same location as provided in provider's identifier mappings submitted to the GRS.

Data descriptions could be improved greatly by further standardising common ways to expose the data, focusing on methods of describing terms that make use of simple techniques such as relying on common vocabularies, and presenting data in data formats that attach further semantic meaning to terms. Such a development should be done in a gradual manner, where users of legacy features see no change, but data provider services come increasingly available for automatic consumers, describing data using well described data terms. Semantic Web technologies are already partly being applied to this problem, for instance in the form of a Resource Description Framework (RDF) [50] version of the MIxS checklists. Similar techniques can be used to describe data provider records. The Genomic Standards Consortium could play a key role in further standardizing access to this data, in a way that is compatible with a Linked Data web.

5 Sequence Filtering and Ranking

5.1 Introduction

Since the 1980s the 16S rRNA gene has played a central role in the systematics of the prokaryotes [23, 25, 51, 52]. Because of this central role, it is routinely sequenced in studies, and recent sequencing efforts have endeavoured to generate its sequence for most of the missing type species [53]. As a result, the gene has been sequenced for almost all of the bacterial and archaeal type strains, resulting in a broad coverage of the known prokaryotic diversity in the public databases of the International Nucleotide Sequence Database Collaboration (INSDC, `www.indsc.org`). The type strains themselves are deposited in at least two culture collections in two different countries preferentially located in different parts of the world, as a requirement for describing novel species or subspecies [54].

Over time, living cultures of the same type strain get distributed across multiple researchers and culture collections around the world, and multiple instances of the 16S rRNA gene of the same type strain are sequenced and deposited in the INSDC databases. Figure 5.1 shows a schematic overview of this spread of cultures and sequence records for the *Paenibacillus polymyxa* type strain. StrainInfo finds over 1,700 INSDC sequence records linked to a

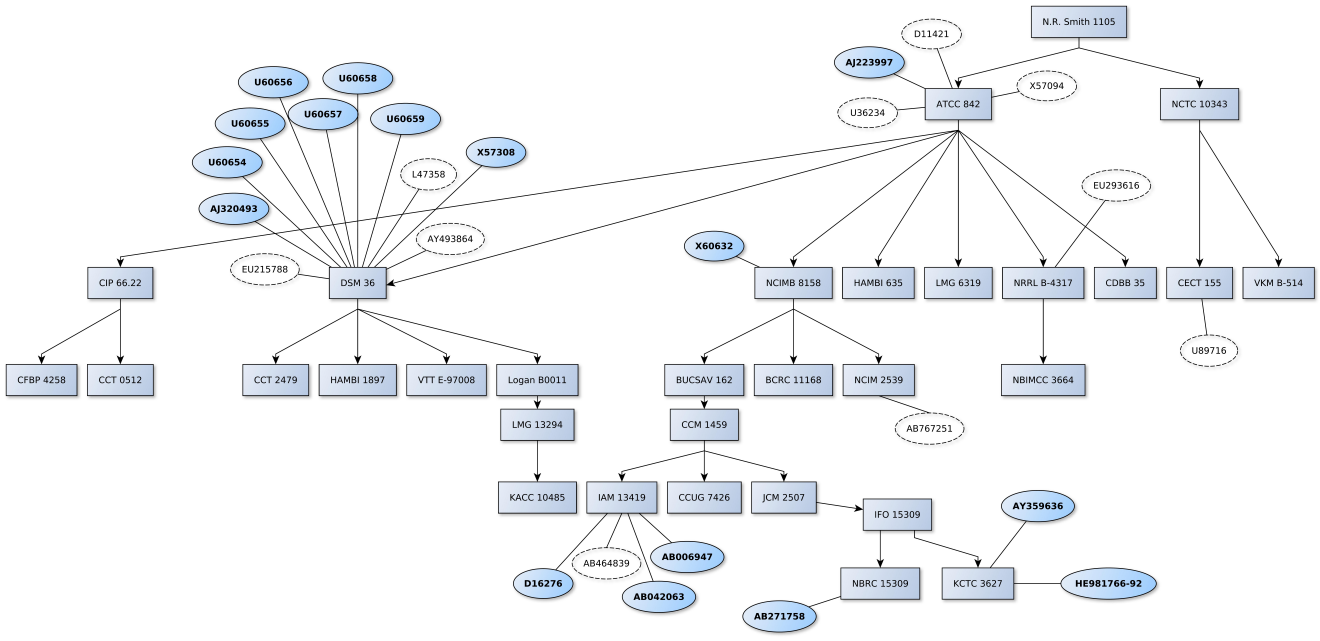


Figure 5.1: Culture exchange history and available INSDC sequences for the *Paenibacillus polymyxa* type strain. Boxes indicate strain numbers assigned to cultures of the type strain held in public culture collections around the world, and arrows indicate transfers of cultures from one collection to another. Accession numbers of INSDC sequence records whose DNA is extracted from a culture of the type strain are indicated by ellipses connected to the strain number of the culture with a solid line. Filled ellipses with solid borders point out (partial) 16S rRNA gene sequences.

strain number of this type strain. The majority of these records result from a genomic survey to reconstruct the draft genome of the P. polymyxa type strain [24] and have been left out of the figure. It is, however, worth noting that 41 of the remaining 52 sequence records contain partial or complete sequences of the 16S rRNA gene with DNA extracted from different cultures of the type strain: DSM 36T (AJ320493, U60654-U60659, X57308), IAM 13419T (AB006947, AB042063, D16276), KCTC 3627T (AY359636, HE981766-HE981792), NBRC 15309T (AB271758) and NCDO 1774T (X60632). This situation where multiple records containing the 16S rRNA gene sequence of the same type strain are available from the public sequence databases is quite common, as can be seen from Figure 5.2.

After the first complete genome of a free-living bacterium was sequenced in 1995 [55], many more whole-genome sequences of Bacteria and Archaea have been deposited in the public sequence databases. Large sequencing projects such as the Human Microbiome Project [56], Genomic Encyclopedia of Bacteria and Archaea (GEBA) [19, 20] and the Microbial Earth Project (`www.microbial-earth.org`) have set out to fill remaining gaps by sequencing the complete genomes of known bacterial and archaeal type strains. All these undertakings lead to multiple sequences of the 16S rRNA gene becoming available for a single type strain.

Some Bacteria and Archaea only have a single rRNA operon, but it is common to find organisms with a higher number of operons [58]. These operons are not always exact copies but generally differ in a modest number of positions, typically less than 15 in the case of the 16S rRNA gene [59, 60]. Although recent analysis [61] revealed more substantial intragenomic rRNA operon variation for human disease associated *Borrelia afzelii*, and the extremophilic prokaryotes *Haloarcula marismortui* and *Thermoanaerobacter tengcongensis*, 16S rRNA gene sequences are still widely accepted and used as phylogenetic markers, especially considering that a study over all sequenced genomes around the same time showed significant intragenomic variability (more than 2%) for only 2% of the examined type strains [60]. The operon variation reported by the authors of [61] also turned out unevenly distributed over the complete length of rrn cistrons and is speculated to be functionally related to adaptations towards unusual ecological conditions. Apart from the expected natural variation in 16S rRNA gene sequences, other differences can be observed in the sequence records containing the 16S rRNA gene of the same strain. While the INSDC enforces a minimum of metadata to be provided when depositing sequences

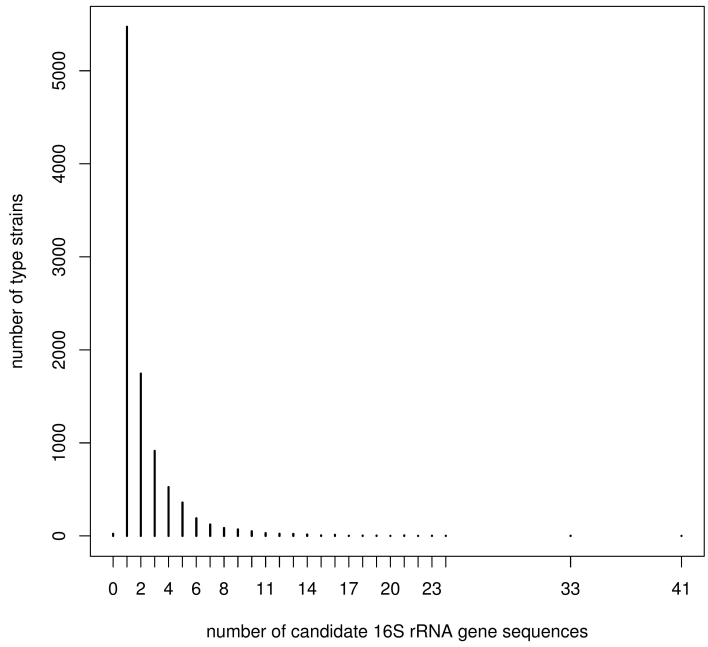


Figure 5.2: Histogram showing the distribution of the available 16S rRNA sequences in the INSDC databases for each bacterial or archaeal type strain. A majority of type strains only has a single 16S rRNA gene sequence, but almost 40% of the type strains have two or more.

in the public repositories, no quality restrictions are in effect for the sequences themselves [45]. As a result, we can find anything in between short partial sequences of the 16S rRNA gene up to full-length sequences. Other confounding factors are badly assembled sequences that contain large gaps filled with Ns or other junk in the form of various vectors at the start or end of the sequence. As the INSDC databases have limited quality control, additional rRNA gene databases such as the Ribosomal Database Project [47, 62], Greengenes [63], EzTaxon [64] or SILVA [31] have been built on top of them to improve their quality by filtering out poor quality sequences or providing quality measures,

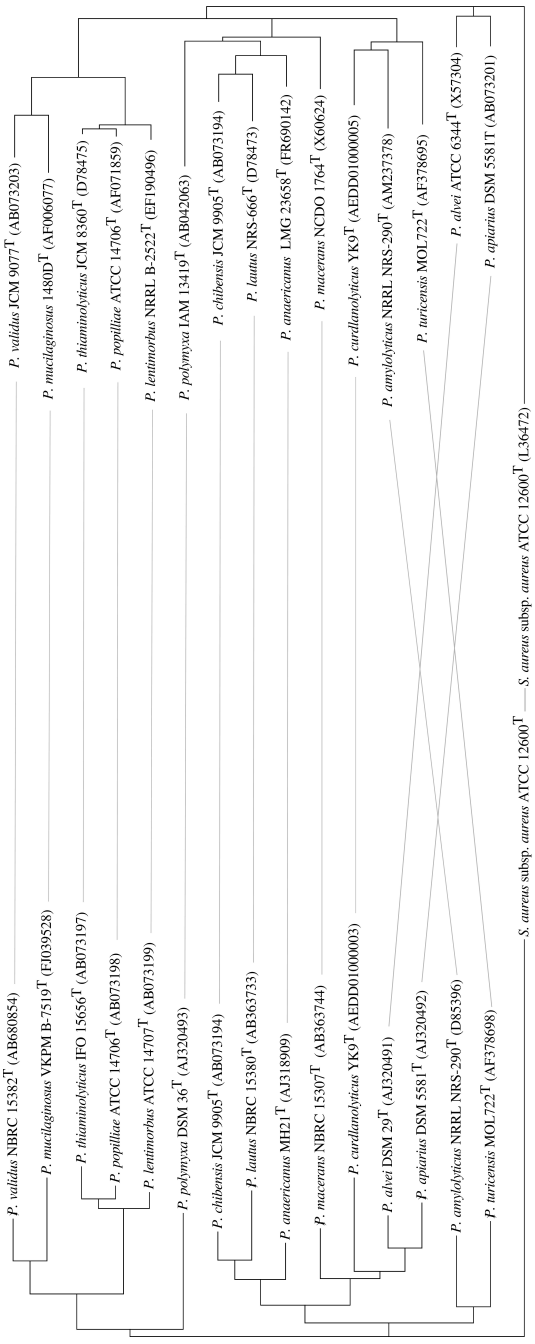


Figure 5.3: Two phylogenetic trees inferred from the same fifteen species in the genus *Paenibacillus*. Species with the highest number of 16S rRNA gene sequences available in the INSDC databases were selected. Left tree constructed from high-quality sequences selected using the ranking techniques discussed in this paper. Right tree constructed from random selection of sequences among the candidate 16S rRNA gene sequences. The tanglegram generated by Dendroscope [57] shows structural differences when comparing both trees.

update or add metadata, and provide additional features such as precomputed high-quality alignments or an entire analysis workbench.

The 16S rRNA gene is universally present and evolutionary conserved, with certain regions exhibiting a higher variability than others resulting in an overall sequence that serves as evolutionary backbone and molecular clock in phylogenetic studies [65, 66]. In addition, large metagenomics project such as the Human Microbiome Project [67] and the Earth Microbiome project [68] use 16S rRNA gene sequences to measure diversity and strongly rely on 16S rRNA reference databases for identification purposes [69]. Given the varying quality of 16S rRNA gene sequences present in the public databases, great care should be taken to only rely upon high-quality sequences when setting up phylogenetic studies or metagenomics analysis. Therefore 16S rRNA gene sequences extracted from the INSDC databases should be screened, leaving out sequences of poor quality [52]. If a representative 16S rRNA gene sequence must be chosen for a particular type strain or other reference strain, all candidate records should be collected and the one of the highest quality should be selected for further analysis. Ideally the entire process should be repeated each time the study is carried out, as sequences are continuously added or updated in the INSDC databases. Since such a screening might be regarded as complex and time-consuming, in practice this step is often ignored or only performed superficially.

As is well-known to practitioners in the field [66], a high-quality selection of 16S rRNA gene sequences is fundamental to the quality of any phylogenetic inference based on it. Figure 5.3 illustrates the importance of this selection. We have inferred two phylogenies for a selection of fifteen species of the genus *Paenibacillus*. In each case, we have selected from the INSDC databases a representative 16S rRNA gene sequence for each species among the sequences that are available from its type strain. In generating the left tree we have carefully chosen the candidate 16S rRNA gene sequence to be of high quality, whereas for the right tree we have randomly picked a sequence among the available candidates. The latter approach generally selected sequences that were shorter and thus carry less information for phylogenetic reconstruction to work with. In comparing the trees, we clearly see that both choices lead to different estimates of the evolutionary distance between the species included in this experiment, ultimately leading to tree topologies showing structural differences. Evolutionary analysis based on one tree might thus fundamentally disagree with a similar analysis based on the other in extreme cases. As such,

this experiment highlights the importance of using high-quality sequences in phylogenetic studies.

The All-Species Living Tree Project (LTP) attempts to remedy this by selecting a single representative 16S rRNA gene sequence for every bacterial and archaeal type strain [60, 70]. Therefore, a curator chooses the most representative sequence manually, partly by taking into account the quality measures provided by the SILVA database project, as well as strain information contained in a list of candidate sequences collected from StrainInfo. All data is quality checked and enriched with metadata, and a high-quality alignment is provided to researchers. While LTP is an invaluable research tool, it has some drawbacks. First of all, it requires keeping abreast of the changes in over 10,000 type strains, resulting in slow release cycles. It is expected that once a choice has been made for a particular type strain, it is not re-evaluated in future releases unless some user points out that a better choice might be available. Secondly, the selection criteria are based on expert knowledge and are not made explicit in the resulting data files. There is no way to check what sequences might have been missed, or why certain sequences have been selected over others. Finally, because of its time consuming nature this curation procedure – although very reliable – scales poorly when it comes to extending it to non-type strains or other housekeeping genes. It is expected that in the next decades to come, a vast number of new Archaea and Bacteria will be discovered and described with 16S rRNA gene sequences as a backbone for classification. It is estimated that the real bacterial diversity is far beyond the known diversity[71], which will render the manual curation approach intractable in the future.

StrainInfo has set out to design and build a proof of concept that completely automates the process of selecting from the INSDC databases a high-quality 16S rRNA gene sequence for each bacterial and archaeal type strain. The recommendations resulting from this pipeline were evaluated against the expert selection made in the All-Species Living Tree Project and are available from the strain and taxon passports on StrainInfo in a dynamic way, because they are based on daily updates of INSDC sequences that are automatically linked to cultures of type strains. This validates the automated procedure for extension to non-type strains and even to other housekeeping genes.

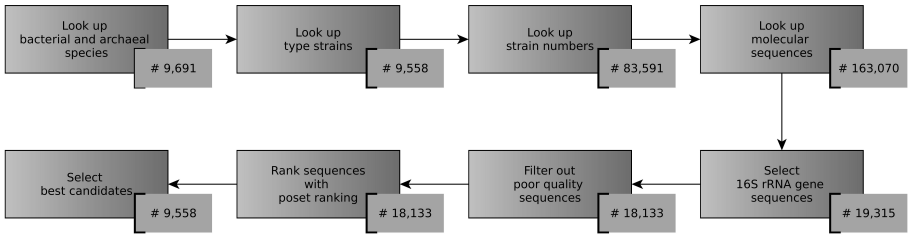


Figure 5.4: Schematic overview of the automated pipeline used by StrainInfo to select high-quality 16S rRNA gene sequences for each bacterial and archaeal type strain. Numbers indicate the number of records that result after each step in the process.

5.2 Materials and Methods

The process of filtering and ranking 16S rRNA gene sequences according to their quality can be divided into a few distinct phases, from the data collection to the ranking itself. A schematic overview of the different phases in this pipeline is shown in Figure 5.4. The type strain of *Paenibacillus polymyxa* [72] will be used as a running example to illustrate the methods involved in each individual phase in the automated selection process. This process consists of data collection of 16S rRNA gene sequences through the use of type strain information, filtering using pre-defined criteria, and ranking of sequences using poset ranking, a multi-criteria ranking method that results in the choice of a single recommended high-quality 16S rRNA gene sequence.

5.2.1 Data collection

In the data collection phase, equivalent strain numbers are used to collect a list of 16S rRNA gene sequences of the same type strain. For species or subspecies for which a neotype strain was accepted by the Judicial Commission of the International Committee on Systematic Bacteriology, this strain was used instead of the type strain. Gene sequences were obtained from the INSDC through the EMBL Nucleotide Sequence Database public FTP servers [42], for EMBL Nucleotide Archive release 104. They were consequently updated using the incremental update files up to August 12, 2013, at which time the tests reported in this manuscript were run.

StrainInfo is used to link species to their type strains and strains to their sequences. These links are accomplished through text mining of various sources such as the List of Prokaryotic Names with Standing in Nomenclature [11], Biological Resource Center (BRC) catalogs, data exports from BRCs in Microbiological Common Language (MCL) format [14], and the sequence records themselves.

Information on species and sequences was imported from the latest LTP release at the time of writing (LTP release 111) available from the LTP project website[1]. Data from this release were exported using the ARB software package [73] and imported into a local database for comparison. Almost all species in LTP release 111 can be linked to a type strain using StrainInfo, as seen in Figure 5.4.

Once a type strain is found, it can be linked to a set of 16S rRNA gene sequences. Feature annotations are used to extract 16S rRNA genes from published sequences where possible, which also allows for extracting the genes from partial or complete published genomes. When no such annotations are present, the description field is scanned for the strings '16S' or 'small subunit'. When multiple identical sequences are available from a genome, only a single copy is retained. Some of these sequences have been submitted as their reverse complement, while not being annotated as such. Therefore all input sequences are checked by the V-REVCOMP software [43] before filtering is done and criterion scores are calculated. While a majority of all bacterial and archaeal type strains has only a single sequence deposited in the INSDC databases, 38.5% of the type strains have two or more sequences. For these sequences an automated ranking procedure will be executed to select a high-quality candidate.

Table 5.1a illustrates the results of this data collection process for the type strain of *P. polymyxa*, for which currently over fifty different (spellings of) strain numbers are known in StrainInfo. Each strain number corresponds to a different subculture of the species, which are in principle clones and can all be used as source material for molecular sequencing. INSDC sequence records are linked to their corresponding strain number by examining the strain annotation provided in many of these records. In the case of *P. polymyxa*, five of its strain numbers were found to have 16S rRNA gene sequence records linked to them.

[1] Data sets of the Living Tree Project can be found at `www.arb-silva.de/projects/living-tree/`

accession	strain	l	a (%)	h (%)	s
filtered sequences					
U60656	DSM 36^T	347	0.0	0.00	NA
U60657	DSM 36^T	347	0.0	0.00	NA
U60658	DSM 36^T	347	0.0	0.00	NA
U60659	DSM 36^T	347	0.0	0.00	NA
U60654	DSM 36^T	347	0.0	0.00	NA
U60655	DSM 36^T	347	0.0	0.00	NA
AB006947	IAM 13419^T	279	0.0	0.00	NA
ranked sequences					
D16276	IAM 13419^T	1,491	0.0	0.07	95.68
AB042063	IAM 13419^T	1,504	0.2	0.07	95.77
AJ320493	DSM 36^T	1,521	0.0	0.07	96.11
X57308	DSM 36^T	1,373	14.3	0.07	81.68
AB271758	NBRC 15309^T	1,479	0.3	0.07	95.85
X60632	NCIMB 8158^T	1,425	3.1	0.07	93.06
AY359636	KCTC 3627^T	1,438	0.0	0.07	95.76

(a) Attribute values

accession	strain	C_l	C_a	C_h	C_s
D16276	IAM 13419^T	1.0	1.000	1.0	0.9864
AB042063	IAM 13419^T	1.0	0.998	1.0	0.9873
AJ320493	DSM 36^T	1.0	1.000	1.0	0.9909
X57308	DSM 36^T	0.8	0.857	1.0	0.8421
AB271758	NBRC 15309^T	1.0	0.997	1.0	0.9882
X60632	NCIMB 8158^T	1.0	0.969	1.0	0.9594
AY359636	KCTC 3627^T	1.0	1.000	1.0	0.9872

(b) Criterion scores

Table 5.1: Attribute values and associated criterion scores for all 16S rRNA gene sequences of the *P. polymyxa* type strain. Table a shows the accession number of each sequence and the strain number of the culture it was derived from, with values for the length (l), the percentage of ambiguous base pairs (a) and the homopolymer percentage (h) in these sequences, and the average similarity of each sequence to the other ones listed (s). Table b lists corresponding criterion scores C_l to C_s for sequences that were not filtered out.

5.2.2 Filtering

Using low-quality genetic sequences as input to the ranking algorithm may adversely affect its result. Excluding them from criteria calculation also allows for the construction of criteria based on the relationship between several sequences, such as in the average similarity criterion defined below. Therefore, these sequences should be filtered out before undertaking criteria calculation and ranking, if it can be ascertained that it is truly unnecessary to take them into account. As a first step, identical 16S rRNA gene sequences within the same genome are only considered once, since only one representative needs to be taken into account.

Some of the common problematic submissions are genetic sequences of only a small section of the 16S rRNA gene. While useful within some phylogenetic contexts, they are preferably not used if a longer, and hence better, sequence exists. Moreover, since these are short reads, they are more likely to contain few read errors and can appear of deceptively good quality, although there are longer sequences with higher phylogenetic content available that a researcher would prefer.

Table 5.1a illustrates the problem for the *P. polymyxa* type strain. In this case there are two groups of sequences. The first one contains very short sequences, while the second one contains the longer sequences that a researcher would normally use. Unfortunately, the short reads are so small that sequence errors are unlikely, which makes comparison to significantly more informative longer sequences difficult. Hence, the short sequences would never be used in a general phylogenetic tree which supports them being filtered out for further ranking.

Filtering on sequence length requires establishing a minimum, as well as a maximum size. The appropriate choice of these thresholds highly depends on what a researcher would find acceptable. Since a large manually curated dataset already consists in the form of the LTP, it can be used to establish these thresholds. As can be seen in Figure 5.5, LTP sequences range in size from 1,135 to 2,210 nucleotides. This is used as the minimum and maximum length in the filtering step of the ranking algorithm, except if such an action would reduce the number of available sequences for a type strain to zero.

As is clear from Figure 5.6, filtering out sequences results in a sharp reduction of the number of sequences that need to be ranked. While a higher number of species only have a single sequence left after filtering, making ranking trivial

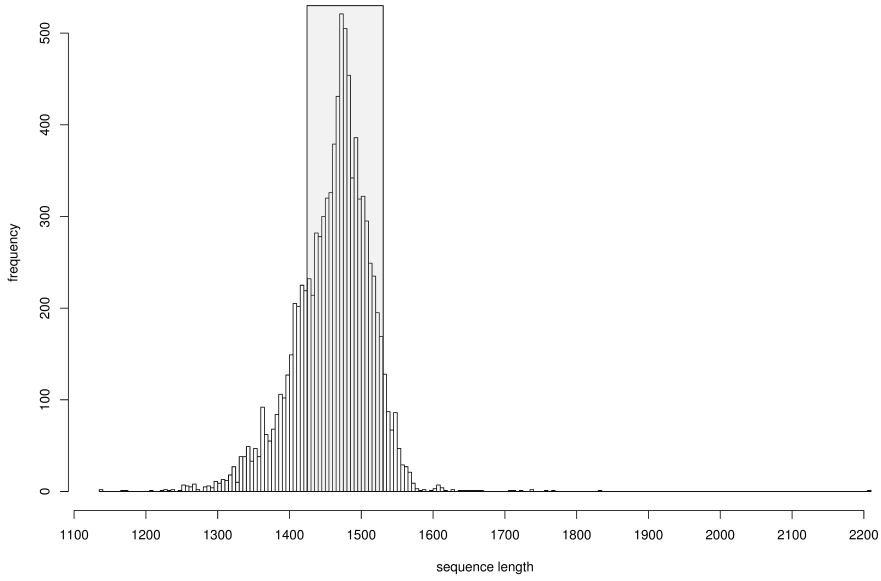


Figure 5.5: Length distribution of LTP 16S rRNA gene sequences. Lengths range from 1,135 to 2,210 nucleotides. The shaded area indicates lengths within one standard deviation of the estimated mode.

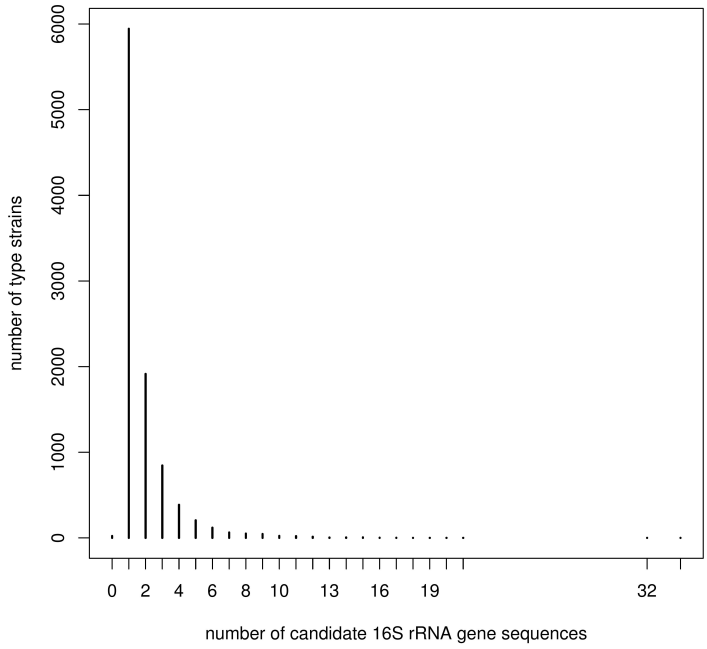


Figure 5.6: Histogram showing the distribution of the available 16S rRNA sequences in the INSDC databases for each bacterial or archaeal type strain, after poor quality sequences have been filtered out. After filtering, a long tail of species remain that have multiple high-quality 16S rRNA sequences.

for these species, a sizable minority has several high-quality sequences left. For these sequences, criterion scores must be determined and ranking must be undertaken.

5.2.3 Criteria

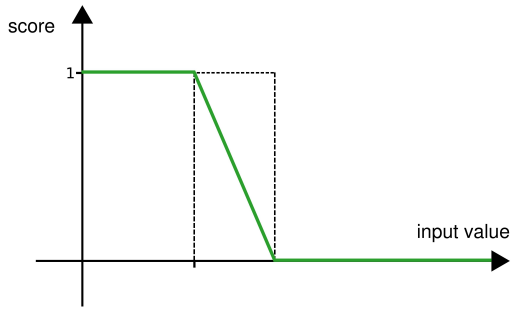
The ranking of candidate sequences that have not been filtered out has to take into account several relevant (and possibly conflicting) attributes. The length, for instance, varies considerably over the available sequences. When comparing two sequences solely according to this attribute, it seems natural

to prefer longer sequences to shorter ones. However, a longer sequence might have been pieced together from a few short reads, separated by long sequences of unknown base pair indicators. This would in turn be reflected by a high value for an attribute that specifies the number of ambiguous base pairs and for which a lower value is clearly preferred. Different concerns such as these lead to different, sometimes conflicting, 'points of view' [74] that have to be taken into account in a final decision on whether a given sequence is preferred to another. For instance, according to Table 5.1a, the sequence with accession number AB271758 is slightly larger than the sequence with accession number AY359636, but it also contains a few more unidentified bases, as evidenced by the ambiguous base pair percentage a. It is not immediately clear which is the better one, as one point of view, which seeks to maximize length, prefers the former, while another point of view, which seeks to minimize ambiguities in the sequence, prefers the latter.

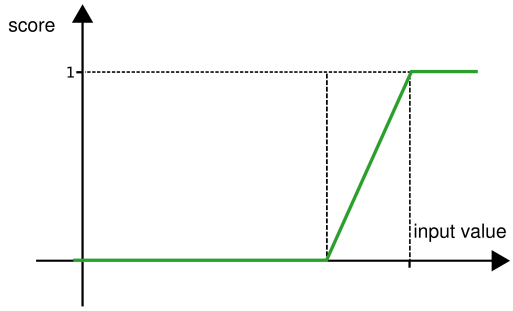
The ordering of sequences under each point of view represents an opinion on which sequences should be preferred to others, and does not necessarily follow the ordering of the sequences under the attribute it is based on. For instance, while longer sequences are generally better for phylogenetic analysis, sequences that are extremely long were likely submitted in error and thus should not take preference over sequences with a length closer to the expected value.

Applying a multi-criteria ranking method such as poset ranking to select the most representative 16S rRNA gene sequence for a given type strain, requires the construction of a criterion for each relevant attribute. A criterion is a scoring function that converts attribute values into real numbers, called scores, so that a higher score implies a higher preference. For simplicity's sake each criterion defined here only uses scores between zero and one, where zero generally denotes the sequence in question is unusable and could be discarded, and one denotes it is perfect according to the point of view. Restricting criteria to a particular range is a convenience for implementing and explaining the algorithm.

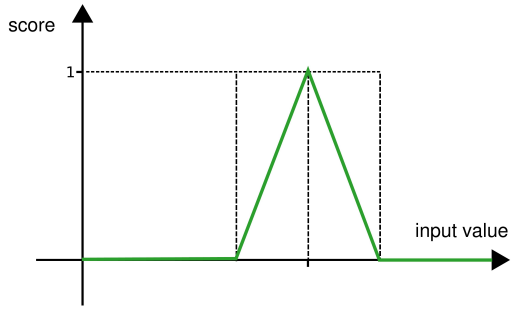
Different preferences for an attribute correspond to different shapes of the scoring function. When it is preferred for sequences to have a very low value for a particular attribute, the scoring function will associate high scores with low input attribute values, such as in Figure 5.7a. Similarly, when sequences with higher attribute values are preferred, the attribute value itself could be the score, if the value is in the interval $[0,1]$. If not, a scoring function such



(a) Minimize input



(b) Maximize input



(c) Target a specific value

Figure 5.7: Three possible scoring functions. The first, Figure a, assigns high scores to the lowest values, thus corresponding to a criterion which seeks to minimize the value for this attribute. Figure b attains the exact reverse, scoring high for the highest values and zero for low values. The last, Figure c assigns high scores to values distributed uniformly around a certain target value.

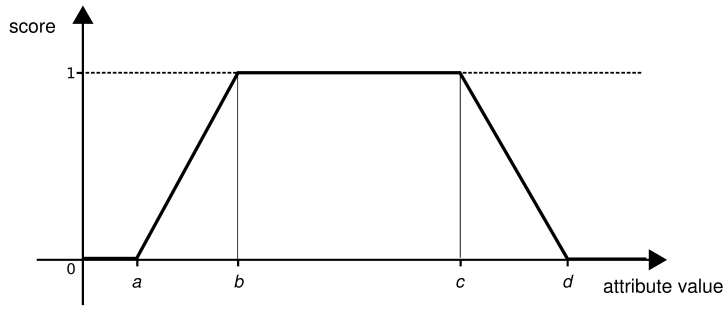


Figure 5.8: Generic parameterized function. Different choices of the parameters a, b, c and d lead to different scoring functions. The configuration above shows a scoring function that associates the highest possible preference (a score of one) with sequences whose attribute value is in the range between b and c.

as in Figure 5.7b could be used. Finally, sometimes there is a certain range of attribute values that is preferred, and any value outside of this range is less desirable.

For the application discussed in this paper, the generic parameterized function in Figure 5.8 provides sufficient options to represent all criteria defined in the Results and Discussion section. This function can be considered the concatenation of five different linear functions: the first takes the constant value zero for all values smaller than a, the second rising linearly from zero to one for values in the range $[a,b]$, the third always one in the range $[b,c]$, the fourth decreasing linearly from one to zero in the range $[c,d]$ and the last zero for all values large than d. Different values of a, b, c and $d$correspond to different scoring functions. The function in Figure 5.8 represents a preference for values in the range $[b,c]$, but by setting a sufficiently low or d sufficiently high, it can also be used to express preferences for high or low attribute values, respectively.

By filtering out sequences in Table 5.1a and choosing criteria that correspond to the sequence's various attributes in Table 5.1a, the poset in Figure 5.9 can

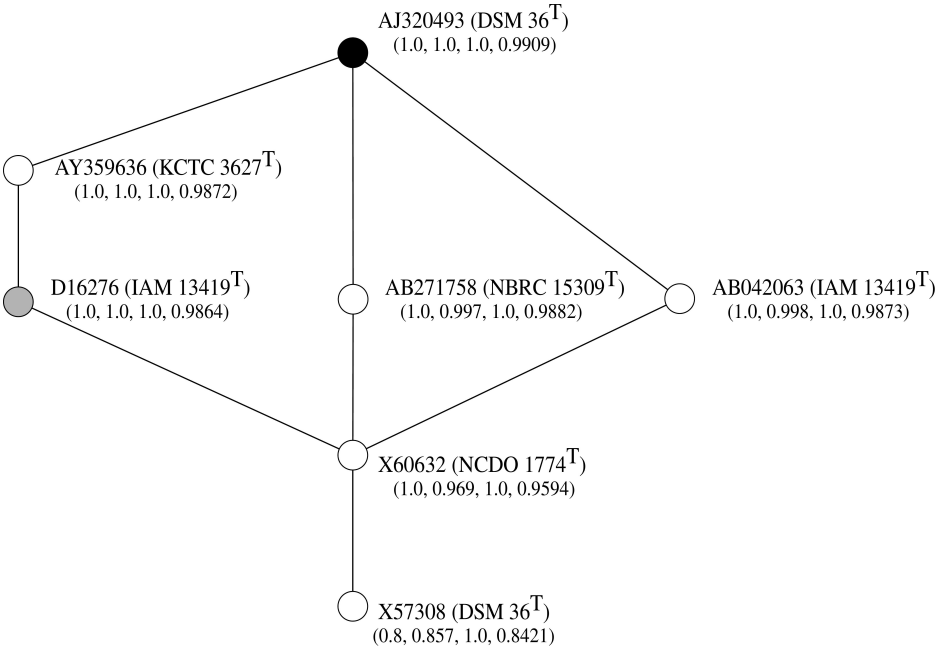


Figure 5.9: Hasse diagram of 16S rRNA gene sequences of the type strain of *Paenibacillus polymyxa*, based on the criteria listed in Table 5.1b. Along with strain number and accession number, the various criteria are listed, in the order of Table 5.1b. The 'best' sequence as determined after linearization is marked in black, while the one selected by LTP is marked in gray. Both are high-quality sequences, but the black one compares slightly better to the gray one in the similarity test. Sequences represented by the nodes located in the lower part of the diagram contain only partial regions of the gene, which causes them to be ranked below the elements representing a full sequence of the 16S rRNA gene.

be obtained. Table 5.1b lists the sequences and the criterion values used in its construction. The Hasse diagram gives a clear overview of the relationship between all remaining 16S rRNA gene sequences for the type strain of *Paenibacillus polymyxa*. Such a representation is much more readable than simply listing the values of all parameters, such as in Table 5.1b. It is also the basis of the ranking algorithm, as good sequences will tend to rise to the top of the graph. In this example, the linearization step explained below is not strictly necessary, as there is a single best 16S rRNA gene sequence.

Generally the criteria considered here favour long and (nearly) complete sequences. The 16S rRNA gene is also quite short and unlikely to include homopolymers. Quality scores based on similar observations are already defined in other databases such as SILVA [31], where they are normalized and combined into a single quality score in an ad-hoc manner. This provided the inspiration for the length, ambiguity and homopolymer criteria defined below.

statistic	*N*	*Est. Mode*	*St. Dev.*	*Min*	*Max*
length	21,369	1,299.849	416.037	5	4,000
ambiguities	21,369	0.195	1.207	0.000	44.200
homopolymers	21,369	0.310	0.396	0.000	33.333

Table 5.2: Summary of attribute statistics for all 16S rRNA gene sequences that could be associated with a type strain of known species in the latest LTP release. Modes were estimated using the R modeest package, version 2.1.

statistic	*N*	*Est. Mode*	*St. Dev.*	*Min*	*Max*
length	9,389	1,477.48	53.028	1,135	2,210
ambiguities	9,389	0	0.211	0.000	2.000
homopolymers	9,389	0.266	0.185	0.000	1.945

Table 5.3: Summary of attribute statistics for all LTP 16S rRNA gene sequences that could be associated with a species type strain.

Table 5.2 indicates that sequences vary in length, number of ambiguities and homopolymers. The sequences recommended by LTP, however, form a quality-controlled subset. The distributions of their attribute values (summarized in Table 5.3) are therefore used to construct the different criteria listed below. This construction process amounts to specifying a way of calculating each attribute, and determining thresholds for the generic parametrized function of Figure 5.8, which are given in Table 5.4.

Length criterion

A 16S rRNA gene sequence is expected to have a length of around 1,541 base pairs, the typical size of the gene in *E. coli* [75]. Various species have slightly

differing 16S rRNA gene sizes. Start and end of the sequence, corresponding to primer sequences, are commonly clipped. Sequences that are significantly shorter or longer (see below) than 1,541 base pairs, however, are not reliable. The former do not provide enough phylogenetic information, while the latter may be assigned incorrectly because of flawed annotation. Therefore, minimal and maximal values for the length distribution of LTP sequences, shown in Figure 5.5, correspond to the cut-off point where sequences are considered 'junk' (at point $a = 1,134$, $d = 2,211$), i.e. score zero for their length criterion. Longer sequences should be ranked according to their length, but, since natural variation in length is expected, sequences with a length close to the mode of this distribution should be considered to be of equal quality. For this reason, sequences with length within one standard deviation of the mode (the range $[b = 1,425, c = 1,530]$) all score one. Since the lengths of all candidate 16S rRNA gene sequences of the *P. polymyxa* type strain, except the sequence with accession number X57308, are within this range, their score C_l is 1.0. The score C_l for X57308 is found by linear interpolation of its length on the line between the points $(a = 1,134, 0)$ and $(b = 1,425, 1)$. The equation of this line is trivially determined as $y = 1/291 * (x - 1134)$, yielding a rounded value of 0.8 for C_l.

Ambiguous base pair criterion

Ambiguous base pairs indicate positions that were not sequenced or for which the reading was uncertain. One can state that the more ambiguities occur in a sequence, the harder it will be to use it for phylogenetic or identification purposes, since these ambiguities do not provide sufficient information and must be masked before making any inferences on the basis of alignments.

The probability of a read error, and thus an ambiguous base pair in the final result, increases with the length of the sequence. Because this has already been captured in the length criterion, the number of ambiguities is normalized in dividing it by the length of the sequence, rounding to 1/1,000th. This results in a value between zero and one.

The estimated mode for this attribute within the LTP data set is zero. This low value indicates that the common case for a high-quality 16S rRNA gene sequence is to contain no ambiguities at all, while sequences that do should be penalised. It is a typical example of a scoring function that should always prefer lower input values. Since the calculated ambiguities can be expressed as a percentage, a, b and c in the generic parameterized function can be set

to zero to achieve this effect. This creates a criterion that is one for sequences with zero ambiguities, and decreases linearly with each extra ambiguity.

Homopolymer criterion

Long repeats of the same base, homopolymeric stretches or homopolymers in short, are uncommon within the 16S rRNA gene. The homopolymer criterion should therefore be maximal (i.e. equal to one) for a sequence that does not contain any homopolymers. Homopolymers are counted by the number H_p, which is defined as the number of homopolymeric events divided by the total possible number of such events. We consider homopolymeric events as tetramers that consist of four identical bases. The total number of tetramers is equal to the length of the sequence minus three.

It is to be expected from chance alone that at least some homopolymers occur in a sequence of the length of the 16S rRNA gene. Indeed, a large fraction of sequences in LTP do have a higher value for H_p than zero, as can be deduced from the mode of this attribute's distribution in LTP, 0.266%, which is still much smaller than the estimated 1.56% in a purely random sequence of this size. As a certain (very small) number of homopolymers are common, this must be taken into account when defining the criterion. Thus, following a similar reasoning as for the length criterion, any sequence that scores a value lower than or equal to the mode of the LTP distribution should score one for this attribute. The corresponding values for a, b, c and d are listed in Table 5.4.

Average sequence similarity

Mistakes can, and often do, crop up in the metadata of a gene sequence. Typically such mistakes would be misspelled strain numbers or mistaken species identifications. While StrainInfo does perform error detection and correction on the input data [10], mistaken identifications may still introduce gene sequences from completely unrelated species or genera into the researcher's phylogenetic tree. In addition, sequences themselves might be described correctly, but might be based on contaminated (unrelated) cultures. Contamination is introduced during regular lab work or during transfer of cultures between researchers and/or biological resource centers.

These unrelated sequences will in most cases be clearly identifiable as outliers in the phylogenetic tree and that phenomenon can be used as the basis

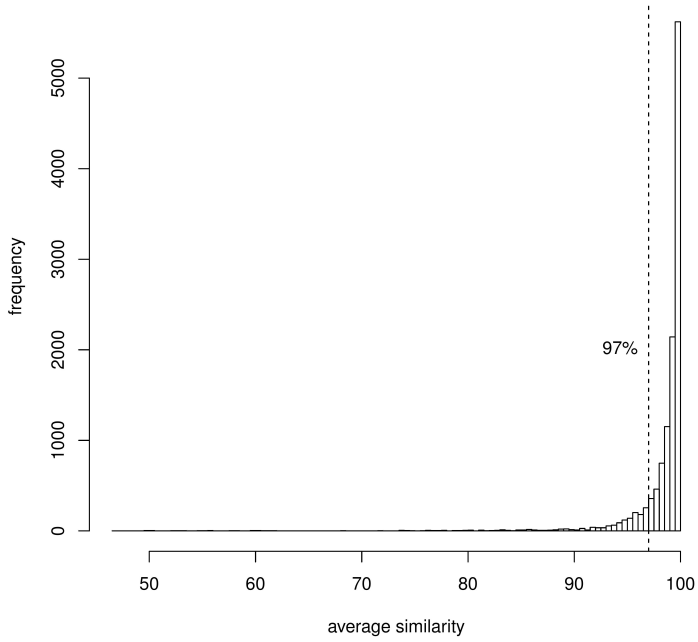


Figure 5.10: Distribution of average similarity over all sequences of type strains for which more than one sequence could be found.

for automatic detection methods. Generally, if two 16S rRNA gene sequences are less than 97% similar [52] they are considered to be from different species. When the alignment and sequences are of high quality, an even more stringent cut-off of 98.7% similarity can be used [76]. It is also clear that when all sequences for the 16S rRNA gene of a particular strain are aligned against each other in pairwise fashion, outliers should clearly have lower similarity values than all others.

The average sequence similarity criterion is therefore based on an unsupervised automated alignment of each candidate 16S rRNA gene sequence of a particular strain to all other candidate 16S rRNA gene sequences, using a modified Needleman-Wunsch algorithm, where alignment is done as normal, but similarity is calculated as the number of identical base pairs in the end

alignment, divided by length of the shortest sequence. After this alignment, the average of these values is taken per sequence. Because of the averaging and automated alignment, the less stringent value of 97% similarity is used as a threshold above which sequences will score one.

Figure 5.10 shows the resulting distribution of similarity values. As expected, most average similarities are very high, but due to outliers, such as contaminants or, exceptionally, intragenomic variation of rrn cistrons, lower averages do occur. If a minority of the sequences revealed for a given strain affect all average similarity values as to reduce them below 97%, comparison according to the criterion is generally unaffected. For this reason, however, the average similarity value cannot be used as a hard cut-off filter in the filtering stage, and is instead used as a separate criterion in the ranking algorithm.

Since the average sequence similarity value is again a percentage, and higher values would be preferred, choosing parameters is straightforward. This time, a is set to zero for convenience and b to the threshold of 97%, c and d are set to 100%. In this way, sequences with average similarity values lower than 97% will be compared on the basis of that similarity value, while sequences with average similarity values above 97% will all score one. A score of one here indicates that they are definitely not outliers, while scores below one may be indicative of problems with the 16S rRNA gene sequence of a particular type strain.

parameter	a	b	c	d
length	1,134	1,425	1,530	2,211
ambiguities	0	0	0	1
homopolymers	0	0	0.266	1
similarity	0	97	100	100

Table 5.4: Generic scoring function parameters that define the scoring functions for the various criteria.

5.2.4 Ranking

Poset Construction

When using only a single criterion, sequences with higher criterion scores are naturally preferred to sequences with lower scores. In this case, a ranking of

the sequences, from most preferred to least preferred, is simply a list of the sequences sorted according to their score. In practice, however, there will be multiple criteria that might conflict with each other: for two sequences X and Y, it can hold at the same time that X is preferred to Y according to the scores of one criterion, and that Y is preferred to X according to the scores of another criterion. In such conflicting cases, the sequences X and Y will simply be considered incomparable, denoted as X ‖ Y. In Table 5.1b, four criteria are given. Here we see that sequences AB042063 and AB271758 are incomparable, since the former scores higher on the ambiguity criterion than the latter, while the situation is entirely reversed for the average similarity criterion. The presence of such incomparable sequences complicates the construction of a single ranking, as the preferred choice between these two sequences is not clear.

A set of sequences, or more generally any set of objects that are evaluated on a number of criteria, can be equipped with a strict partial order relation, denoted as $<$, which specifies that it holds for two objects X and Y that $X < Y$, i.e. Y is preferred to X, if Y scores at least as high on all criteria as X and higher on at least one criterion. By doing this, a partially ordered set (or poset) is obtained. In the poset obtained from the sequences in Table 1(b), for instance, X57308 < AJ320493 and X57308 < AB271758. A poset can be depicted graphically by a Hasse diagram, as illustrated in Figure 5.9. In a Hasse diagram, the sequences are represented by nodes that are connected by edges. An edge is drawn between a node X and a node Y that is drawn higher if it holds that $X < Y$ and that there is no third node Z such that $X < Z < Y$. Note that two sequences are comparable if and only if there is a path in the diagram upwards or downwards from one to the other. When there is no path leading up from a sequence, it is said to be maximal. In other words, a maximal sequence is one for which no better sequence can be found. In Figure 5.9, exactly one sequence is maximal, but there may be (and often are) several incomparable maximal sequences. Figure 5.11 illustrates this with a synthetic example in which both node A and node E are maximal. While node A is comparable to most other nodes, node E is maximal because it is incomparable to all of them.

Linearization and Ranking

A ranking algorithm [77], which ranks the objects of a poset without assigning a common scale or weights to the criteria, is used here to construct a single ranking of all unfiltered candidate 16S rRNA gene sequences from the same

object	criterion C_1	criterion C_2
A	1.0	0.8
B	0.5	0.7
C	0.6	0.6
D	0.6	0.5
E	0.0	1.0

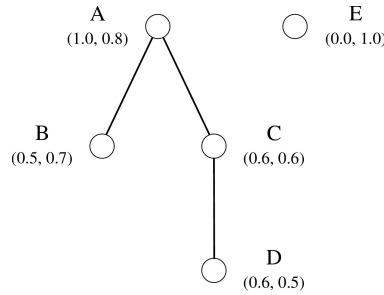


Figure 5.11: An example poset that illustrates the basic elements of a Hasse diagram. Note that two nodes are connected only if one directly covers the other, and the covered node is placed lower in the drawing. Also note that object E is incomparable to the other nodes and that its node thus is not connected to any other node in the diagram.

type strain. In this section, the simple example poset from Figure 5.9 is used in an intuitive illustration of this poset ranking algorithm. Interested readers are referred to [78] for background and [77] for more mathematically rigorous definitions and a description of the ranking software used.

The ranking algorithm first constructs a poset from the sequences to rank by examining their criterion scores, as described above. If there is only one sequence that is considered maximal under the criteria, it can be selected as the best one. But often, not all sequences will be comparable to each other under the criteria as outlined above. Two or more of such sequences may also be maximal, i.e. not topped by another sequence. Since the ranking algorithm is looking for a single representative, a choice has to be made between them.

Poset ranking provides the mechanism for this choice, by solving the more general problem of finding a single, representative ranking of all objects in the poset. Such a ranking places the best sequence or sequences at rank one, the second best at rank two, and so on. There are usually many different possible rankings that are compatible with the partial order of the poset. Such rankings are called linear extensions. In these rankings, an object can only be higher in rank than another if it is higher in the original poset, or if it is incomparable to the other. For instance, in Figure 5.9, any valid linear extension will be a list starting with AJ320493 followed by D16276, and ending with X57408 preceded by X60632. On the other hand, AB042063, AY359636 and AB271758 may be

rank	linear extensions									
1	AJ320493	AJ320493	AJ320493	AJ320493	AJ320493	AJ320493	AJ320493	AJ320493	AJ320493	AJ320493
2	AY359636	AY359636	AB042063	AB271758	AB042063	AB271758	AY359636	AY359636	AY359636	AY359636
3	D16276	D16276	AY359636	AY359636	AB271758	AB042063	AB042063	AB271758	AB042063	AB271758
4	AB271758	AB042063	D16276	D16276	AY359636	AY359636	D16276	D16276	AB271758	AB042063
5	AB042063	AB271758	AB271758	AB042063	D16276	D16276	AB271758	AB042063	D16276	D16276
6	X60632	X60632	X60632	X60632	X60632	X60632	X60632	X60632	X60632	X60632
7	X57306	X57306	X57306	X57306	X57306	X57306	X57306	X57306	X57306	X57306

Table 5.5: List of all possible linear extensions of the poset in Figure 5.9, derived from the sequences of the *P. polymyxa* type strain. Each column is a linear extension of the original poset, position in the column indicates rank 1 through 7. Average ranks for each sequence are: AJ320493: 1.0;. AY359636: 2.6; AB271758: 3.6; AB042063: 3.6; D16276: 4.2; X60632: 6.0; X57306: 7.0.

listed in any order. Table 5.5 lists all possible linear extensions for this particular poset. To every pair of incomparable objects, two different orders on the objects correspond. As a result, a poset has in general many linear extensions. In every linear extension, each sequence has a certain rank. A good approximation of how it compares to other sequences in the poset is the average of its ranks over all linear extensions, which is the rank that is calculated and used by the ranking algorithm to determine the highest ranking sequence.

In order to obtain the average ranks, all possible linear extensions of the poset can be generated as in Table 5.5, and the average rank can be calculated by averaging over all ranks in the linear extensions. In practice, however, the algorithm does not enumerate all linear extensions of the poset, as the exponential behaviour of the number of linear extensions would render this naive approach infeasible even for smaller posets [79]. In order to avoid explicit enumeration, it constructs a data structure called the lattice of ideals, in which efficient graph counting operations allow for direct computation of the average ranks. We mention that several algorithms to approximate the average ranks have been devised for large posets for which the exact computation is infeasible, ranging from a simple formula to obtain a crude estimate of the average ranks to a Markov chain Monte Carlo algorithm. The interested reader is referred to [80], [77] and [79]. If there is a single object with the highest average rank, it is returned as the best (or most preferred) object. If multiple objects share the highest rank, either all can be returned, or a tie breaking routine can be used to arbitrarily choose a single winner.

5.3 Results and Discussion

5.3.1 Finding candidate 16S rRNA gene sequences

Finding all 16S rRNA gene sequence records in the INSDC databases that belong to a given type strain is not always an easy task, as the quality of their annotation varies considerably, from annotations that are extremely complete and clear, to annotations that are too vague or simply missing, especially for older sequence records. The approach described above links 16S rRNA gene sequence records to species through their type strains, using type strain information gathered through text mining from LPSN, existing strain information in StrainInfo, and INSDC sequence record annotations. The advantage of linking at the strain level is that sequence records can be found through their strain annotations, even when other sequence record descriptors, such as the description field itself or the organism name, do not indicate that the sequence record was derived from a type strain. Its most obvious downside is that strain annotations may be missing or mistaken. In addition, StrainInfo relies on Biological Resource Centers for the quality of its strain information, and thus strain information may in some cases be incomplete. In order to better understand the extent of these issues, all species for which the 16S rRNA gene sequence record that is recommended by LTP was not automatically found by StrainInfo were further examined in detail. A complete list of these sequences and the various reasons why they could not be discovered automatically is provided in the next section.

At the time of the comparison, ninety-nine 16S rRNA gene sequence records recommended by LTP were not linked to a type strain in StrainInfo when performing the type strain to sequence lookup process described in the methods section, a reasonably small portion (1%) of the 9691 species names listed in LTP. In the case of four species, this was simply due to a spelling conflict between the species names listed in LTP and the spelling as in LPSN that is followed by StrainInfo. These species names, with spelling used in LTP, are: *Trueperella bialowiezense*, *Listeria rocourtia*, *Bacillus purgationiresistans*, *Desulfobaculum xiamenensis*. The spelling differences were resolved as being synonymous when comparing the StrainInfo data set with the LTP data set.

The remaining cases are instructive for the general types of issues that arise during data integration. For forty-seven 16S rRNA gene sequence records, StrainInfo data could be updated with corrected links to 16S rRNA gene sequences which were either not previously linked to a strain (due to missing

strain annotations in the sequence record and not having linked the sequence to the strain through LPSN, for 11 sequences), not yet recognized as belonging to a type strain (10 sequences), or carrying a mistaken strain annotation (26 sequences). The latter showcases a major downside of relying on these annotations, as their reliability varies from study to study. However, the majority of examined sequences were linked to the correct strain, and several of these sequence records have already been updated to resolve such issues. In future, metadata standards should ensure that providing useful annotations such as source strain information to sequence records becomes best practice [45, 81], allowing the aggregation approach to reduce both false positives and false negatives.

Another 48 sequences recommended by LTP were not retrieved for ranking. For 14 of these sequences, this was caused by the complete absence or low quality of feature annotations, which are used to extract 16S rRNA gene sequences from the sequence records. For about 30 remaining strains, the 16S rRNA gene sequence recommended by LTP does not appear to be derived from the type strain. These apparent errors were reported to the LPSN and LTP curators and should be resolved in subsequent releases. This clearly highlights the complementary aspect of both data processing approaches to sequence retrieval: LTP curators usually track down sequences using publication data, even if there are gaps in the annotations, while strain information used by StrainInfo can highlight inconsistencies and sometimes find a larger array of sequences, if such annotations were correctly provided. In this way, regular quality checks of both data sets can improve the overall quality of both beyond what can be accomplished through either approach on its own.

Fortunately, for most species, it is still possible to find the majority of 16S rRNA gene sequences by relying on the annotations or by doing a full text search on the description field. In the end, only 13 type strains could not be associated to any candidate 16S rRNA gene sequence at all, most of them because of a lack of feature annotations. Partly this problem could and should be solved by calling on the community to improve annotations of strains and sequence features. It is encouraging that this is already part of or is being integrated into the submission pipelines of the INSDC databases.

For the 16S rRNA gene in particular, there also exist expert databases such as the SILVA [31] and RDP [47] databases, which identify additional 16S rRNA gene sequences, presumably with the help of statistical modelling tools such as RNAmmer [65]. By relying on these databases, it should be possible to include

those sequences which lack any reasonable feature annotations or full-text descriptions, especially if these expert databases could be used to augment the annotations made in the INSDC databases.

5.3.2 Missing LTP sequences

The following two lists contain all LTP sequences that, at the time of tests, were not found in the 16S rRNA gene sequence set for their type strain. The first list contains LTP sequences that were manually verified and should be linked to their type strain. The second list contains all sequences that were not linked to a type strain, either because of certain limitations in the use of feature annotations, inconclusive evidence for the type strain or the fact that some of the sequences appear to originate from a non-type strain. Each entry in the below lists includes the species name and sequence accession number as it is found in the database for LTP release 111.

Sequences manually linked to type strain

- Sequences not automatically linked to strain number: these sequences should be linked to the type strain, but generally no evidence was found for the type strain (in sequence record annotations or elsewhere) that could conclusively link the particular sequence to a type strain prior to this analysis.

  Bacteroides galacturonicus DQ497994
  : Manually linked to strain number N6 based on info in Bergey's Manual.

  Ehrlichia ewingii M73227
  : No strain was listed in the sequence record.

  Sulfurospirillum multivorans X82931
  : The strain number DSM 12446 was only mentioned in its Genbank record, not in ENA or DDBJ.

  Methanosaeta concilii CP002565
  : This sequence record mentions three strain numbers: GP6 = ATCC BAA-1996 = OCM 252; the extraction algorithm selection ATCC BAA-1996, but this strain number is not known as a synonymous strain number of the type strain by StrainInfo; the entry does not exist in

the ATCC catalog either; also OCM 252 is not recognized as the type strain (which includes OCM 69).

Weissella kandleri M23038
: Strain number DSM 20593 is mentioned in a comment section, which is not parsed by the strain number extraction algorithm because it is not part of the entry in the ENA; no other strain number was mentioned in the source feature.

Citrobacter youngae AB273741
: Strain number GTC 1314, mentioned in the sequence record, turns out to be a synonym of the type strain according to the equivalence information found in this paper; however, since this strain number was not mentioned in any online catalog of a public culture collection, this information was unknown to StrainInfo; the online catalog of the GTC collection can be found at `http://gtc.jpn.com/`, but it does not contain a reference to the GTC 1314 strain number, nor the type strain of *C. youngae*; the GTC collection is currently not indexed by StrainInfo.

Carboxydothermus pertinax AB573433
: Strain number Ug1 was correctly extracted from the sequence record, but could not be linked automatically to a StrainInfo culture ID since the accession number mentioned by LPSN is oddly formatted (accession numbers of four different clones); only the first one was retained for automatic linking, given the extra information that this is a sequence linked to the type strain.

Hoeflea phototrophica ABIA02000018
: LPSN does not mention this accession number, but an accession number of a 16S rRNA gene sequence of this strain that is only 800bp long.

Thermolithobacter ferrireducens AF282253
: Strain number mentioned in sequence record is KA2 (or clone KA2a), whereas the correct strain number from the type strain is JW/KA-2; as a result, this sequence record could not be automatically linked to the type strain and was not found earlier since LPSN mentions another accession number for the 16S rRNA gene sequence of the type strain.

Gallibacterium melopsittaci EU423985
: Not automatically linked to culture, since LPSN mentions two accession numbers, of which only the first one was automatically linked given the contextual type strain information.

Haloarchaeobius iranensis JF293279
: Sequence record not automatically linked to culture, since LPSN mentions two accession numbers (two different rRNA operons), of which only the first one was automatically linked given the contextual type strain information.

- Strain numbers which were not automatically recognized as part of the type strain. StrainInfo uses an integrative learning algorithm, which sometimes has not yet seen enough information to link certain strain numbers directly to the type strain. The following entries were all linked manually given publication or other manually verified data.

  Nonomuraea jiangxiensis FJ418910
  : Strain number FXJ1.102 could not be uniquely resolved at time of parsing.

  Salinactinospora qingdaonensis GU253338
  : Strain number CXB832 could not be uniquely resolved at time of parsing.

  Campylobacter sputorum subsp. *sputorum* X67775
  : Linked to type strain based on information in the sequence record description field.

  Bacillus persepolensis FM244839
  : Strain number listed as "HS-136" was linked to strain number HS136.

  Eubacterium cellulosolvens X71860
  : Strain number ATCC 43171 linked to type strain.

  Bacillus chitinolyticus AB045100
  : Linked to type strain based on information in sequence record: HSCC 596T (IFO 15660T).

  Bacillus pabuli AB045094
  : Linked to type strain based on information in sequence record: HSCC 492T (NRRL NRS-924T).

Segniliparus rotundus AY608918
: Linked to type strain based on information in sequence record: CDC 1076.

- Sequence records not linked to type strain because of an erroneous INSDC source modifier: this list reflects a set of sequence records for which the strain or culture collection information in INSDC is partly or wholly in error.

Actinomadura hallensis DQ076484
: Sequence record lists non-type strain number KCTC 9992 next to type strain number H647-1.

Methylococcus mobilis HQ676599
: The `Culture_collection` source modifier "NCCB<NLD>:77028" does not appear to conform to INSDC guidelines. The annotation should normally read "NCCB:77028".

Verrucosispora lutea EF191199
: Erroneously linked to the strain number YIM 1013, since corrected to read YIM 013.

Actinomadura longispora AF114809
: "Erroneously linked to NRRL-B 116116, which should read NRRL B-16116."

Bradyrhizobium yuanmingense AF193818
: Strain number "B071" listed in sequence record does not match strain "CCBAU 10071" listed in original species publication.

Corynebacterium auris X81873
: Erroneously linked to DZZM 328, should read DMMZ 328.

Corynebacterium accolens AJ439346
: Erroneously linked to ATCC 49724, should read ATCC 49725.

Desulfobulbus marinus M34411
: Erroneously linked to DSM 3380, should read 3pr10.

Legionella quateirensis Z49732
: Erroneously linked to NCTC 12370, should read NCTC 12376.

Mycobacterium hiberniae X67096
: Erroneously linked to ATCC 9874, should read ATCC 49874.

Bacteroides helcogenes AB200227
: Erroneously linked to the strain number JCM 6927, since corrected to read JCM 6297.

Mycobacterium szulgai X52926
: Erroneously linked to ATCC 25799, should read ATCC 35799 (mistake was also made in publication associated with sequence record).

Neisseria gonorrhoeae X07714
: Erroneously linked to NCTC 83785, should read NCTC 8375.

Plantibacter auratus AB177868
: Erroneously linked to IAM 18417, should read IAM 14817; it is highly questionable that both sequences of this strain are correct, as they are only 93% similar.

Caulobacter vibrioides AJ009957
: Erroneously linked to VKM-B-1496, should read VKM B-1496.

Salinivibrio costicola X74699
: Erroneously linked to ATCC 35508, should read ATCC 33508.

Salinivibrio costicola subsp. *costicola* X74699
: Erreoneously linked to ATCC 35508, should read ATCC 33508.

Streptococcus gallolyticus subsp. *pasteurianus* AJ297216
: Erroneously linked to CIP 105070, should read CIP 107122.

Streptococcus pasteurianus AJ297216
: Erroneously linked to CIP 105070, should read CIP 107122.

Streptomyces aburaviensis AY999886
: Erroneously linked to AS 4.1869, should read AS 4.1469.

Shinella zoogloeoides AB238789
: Erroneously linked to ATCC 19263, should read ATCC 19623.

Thermobacteroides leptospartum AF266461
: The strain number "ATCC 35414" listed on EMBL is the type strain of a different subspecies. It is unclear whether this is an author error, or an error in the EMBL record.

Vibrio costicola X74699
: Erroneously linked to AS ATCC 35508, should read ATCC 33508.

Sporosarcina pasteurii HQ676600
: The `Culture_collection` modifier "NCCB<NLD>:48021" does not appear to conform to INSDC guidelines. The annotation should read "NCCB:48021".

Methylophaga marina X95459
: Erroneously linked to DSM 5989, should read DSM 5689.

Sequences not linked to type strain or not retrieved by algorithm

The sequences in the below list can be further subdivided into two groups: sequences not retrieved because no evidence could be found that they were derived from the type strain, and sequences not retrieved because of limitations in the algorithm, mainly because of the use of 16S rRNA gene feature annotations, which are not always present, or do not always indicate the particular type of rRNA that a feature describes.

- Sequence which could not be automatically linked to a strain number.

  Burkholderia contaminans GQ397111
  : Strain number I29B was extracted from the sequence record, but not recognized by StrainInfo as a strain number of the type strain; given the information in one of its publications [82], it is doubtful whether this sequence is from the type strain; this paper states that the I29B strain was isolated from soil, whereas the type strain was isolated from sheep with mastitis (milk).

- Complete genome sequences without 16S feature annotations: these sequence records contain annotations for protein coding genes, but its authors appear to have elected not to provide rRNA feature annotations, which makes it impossible to use these sequences when using feature annotations to extract 16S rRNA gene sequences.

  Caldanaerobacter subterraneus subsp. *pacificus* ABXP01000185
  : No feature annotations where provided with this large contig.

  Fusobacterium nucleatum subsp. *nucleatum* AE009951
  : No 16S rRNA gene annotation found.

  Clavibacter michiganensis subsp. *sepedonicus* AM849034
  : No 16S rRNA gene annotation found.

- Sequence records for which feature extraction heuristic has failed, either because no annotation was provided and the description of the sequence was not clear, or because provided annotations do not indicate whether the feature contains the 16S rRNA gene or not.

  Bacillus thermoamylovorans L27478
  : Annotated only as "ribosomal RNA" or "rRNA".

  Desulfacinum infernum L27426
  : Annotated only as "ribosomal RNA" or "rRNA".

  Desulfovibrio longreachensis Z24450
  : Annotated only as "ribosomal RNA" or "rRNA".

  Halanaerobium lacusrosei L39787
  : Annotated only as "ribosomal RNA" or "rRNA".

  Haloferax prahovense AB258305
  : Annotated only as "ribosomal RNA" or "rRNA".

  Mycobacterium insubricum EU605695
  : Contains intragenic spacer.

  Rhabdochromatium marinum X84316
  : Annotated only as "ribosomal RNA" or "rRNA".

  Saccharopolyspora spinosporotrichia Y09571
  : Annotated only as "ribosomal RNA" or "rRNA".

  Thiothrix unzii L79961
  : Annotated only as "ribosomal RNA" or "rRNA".

  Mogibacterium timidum Z36296
  : Annotated only as "ribosomal RNA" or "rRNA".

  Eubacterium nodatum Z36274
  : Annotated only as "ribosomal RNA" or "rRNA".

  Eubacterium brachy Z36272
  : Annotated only as "ribosomal RNA" or "rRNA".

  Desulfotomaculum thermosapovorans Z26315
  : Annotated only as "ribosomal RNA" or "rRNA".

  Halothermothrix orenii L22016
  : Annotated only as "ribosomal RNA" or "rRNA".

Ensifer medicae L39882
: Annotated only as “ribosomal RNA” or “rRNA”.

- Species which were not listed in LPSN.

  Iphinoe spelaeobios HM748317
  : No cultures of the type strain were deposited in public culture collections.

  Loriellopsis cavernicola HM748318
  : No cultures of the type strain were deposited in public culture collections.

- Species with likely erroneous strain annotations in INSDC records.

  Clostridium stercorarium subsp. *leptospartum* AF266461
  : The strain source modifier lists the type strain of a different subspecies of *C. stercorarium* than indicated in the sequence record description. This is likely due to author error.

  Pseudomonas fulva AB046996
  : This could either be a spelling mistake in the type strain as mentioned in the INSDC sequence record (AJ 2129 should then read AJ 2219), or this sequence is from the type strain of *P. parafulva* according to the strain number, in which case the taxonomic name mentioned in the sequence record is not correct and this is no 16S rRNA gene sequence of the type strain of *P. fulva* as mentioned in LPSN and LTP.

  Legionella londiniensis Z49730
  : Originally this sequence was submitted as being derived from the *L. nautarum* type strain. Its description and organism description have since been adjusted, but one of the strain numbers listed remains that of the *L. nautarum* type strain. It is unclear whether this is an oversight, or whether the original annotation was correct. The opposite situation has occurred on the *L. nautarum* sequence record.

  Legionella nautarum Z49728
  : Refers in its strain description to the *L. londiniensis* type strain, see *L. londiniensis*.

- LPSN and LTP mark a sequence likely not derived from the type strain. In researching the following entries no evidence was found that they belong to the type strain of the indicated species, either because they belong to a non-type strain, or because they belong to the type strain of a different, often related species.

  Mycoplasma verecundum AF412989
  : Annotated as being derived from strain "GIH", which is not the type strain of *M. verecundum*.

  Comamonas terrigena AF078772
  : Non-type strain.

  Corynebacterium auriscanis AJ243819
  : Non-type strain.

  Cupriavidus pauculus AF085226
  : Non-type strain; should be EU024165; in publication for this species, the wrong strain (with accession number) is stated as the type strain.

  Desulfovibrio gigas AY726757
  : Should be linked to DSM 1382; this sequence record has been modified multiple times over time, with modifications on both species name and strain number; see also accession number DQ447183. Both LTP and LPSN list this sequence with the older mentioned species name.

  Desulfovibrio paquesii DQ447183
  : Should be linked to SB1; sequence record has been modified multiple times over time, with modifications on both species name and strain number; see also accession number AY726757. Both LTP and LPSN list this sequence with the older mentioned species name.

  Ectothiorhodospira shaposhnikovii FR733667
  : Non-type strain.

  Ensifer kostiensis AM181747
  : Non-type strain.

  Eubacterium minutum U13037
  : Not from type strain, but from type strain of *E. tardum*.

Geobacter grbiciae AF335183
: Non-type strain; should be AF335182.

Lactobacillus plantarum subsp. *plantarum* AJ965482
: Not from type strain, but from type strain of *L. arizonensis*.

Lactobacillus sakei subsp. *carnosus* AY204889
: Not from type strain, but from type strain of *L. curvatus* subsp. *melibiosus*.

Megasphaera cerevisiae L37040
: Non-type strain.

Oceanospirillum maris AB006763
: Not from type strain, but from type strain of *O. maris* subsp. *williamsae*.

Propionibacterium acidipropionici AJ704570
: Non-type strain.

Pseudomonas parafulva AB046999
: Not from type strain, but from type strain of *P. cremoricolorata*.

Streptomyces abikoensis AY999834
: Not from type strain, but from type strain of *S. ehimensis*.

Streptomyces netropsis EF178671
: Not from type strain, but from type strain of *S. distallicus*.

Thalassomonas viridans AJ294747
: Non-type strain; should be AJ294748.

Geobacillus thermodenitrificans subsp. *thermodenitrificans* AY608961
: Non-type strain.

Gordonia rubripertincta AY995557
: This accession number is linked to the type strain of *Rhodococcus corallinus*, which is a heterotypic synonym of *Gordonia rubripertincta*.

Pediococcus urinaeequi D87677
: Non-type strain.

5.3.3 Filtering

It is possible to implement more filters on the basis of other attributes. Choosing thresholds for most attributes is not easy, however, as filtering must not

remove useful sequences. Furthermore, when different filters are configured, they will typically filter different subsets of sequences, possibly occasionally filtering all available sequences, even though it is considered necessary to retain at least one 16S rRNA gene sequence. Determining which sequence to keep in that case is an example of reconciling conflicting points of view, and is left to the poset ranking algorithm for exactly that reason. Other projects handle filtering differently, and remove sequences with too many ambiguous base pairs [83] in addition to a length cut-off. This case is handled naturally by the poset ranking algorithm. The problem of presence of chimeras and sequences with large embedded vectors is handled partly by the similarity criterion. In future versions, we plan the introduction of a chimera filter, such as pintail [84] or Bellerophon [85], and the use of universal alignments provided by expert databases to further improve filtering. By using the latter to align sequences, it should also be possible to adjust for broken annotations, which may sometimes include residues into the feature which are not part of the gene, although we have no evidence that in practice this is a problem for the ranking algorithm. By excluding obviously low-quality sequences however, it should be possible to further improve ranking results.

5.3.4 Adding more criteria

The quality of the poset ranking result relies heavily on the choice of criteria. The more criteria are used, the more likely it is that two objects of the constructed poset are incomparable. More incomparable objects result in more linear extensions to evaluate, which will result in increased runtime. Selecting criteria that work well together is thus critical. One of the main pitfalls is the use of heavily correlated criteria. Indeed, when two criteria are correlated, the orderings according to one criterion will be very close to the other, but since the values are so close together, it is likely that reordering of sequences under the other criterion happen purely due to natural variation. Or in the worst case, a negative correlation will mean the ordering is the same but reversed. Every such reordering makes ranking harder, and does not add real useful information. It is therefore wise to eliminate any redundant criteria, since they only serve to lower the quality of the final result.

Table 5.6 shows the mutual correlation for each combination of intrinsic criteria, for both the original attribute values (for instance for length this is the actual length of a sequence) and the criterion scores (a value between zero and

attribute	*length*	*ambiguities*	*homopolymers*
length	1.00	-0.03	0.06
ambiguities	-0.03	1.00	-0.02
homopolymers	0.06	-0.02	1.00

(a) Correlation between original attributes

criterion	*length*	*ambiguities*	*homopolymers*
length	1.00	0.06	0.06
ambiguities	0.06	1.00	-0.01
homopolymers	0.06	-0.01	1.00

(b) Correlation between criterion scores

Table 5.6: Correlations between criteria and between the attributes they are based on, computed over all available sequences

one) used in the current ranking. Correlation is very low for all values, which suggests that none of the current criteria can be considered redundant.

While the four criteria used here are the only ones currently tested, many other criteria could still be devised. These could include the age of the sequence, the results of pintail tests and checks for unwanted vectors residing in the sequence, although the latter two are likely more relevant to filter out low-quality sequences entirely. Furthermore, adding a universal alignment of all 16S rRNA gene sequences from expert databases, as suggested above, could also provide new possibilities for criteria, such as ones that reflect the secondary structure conformity of the 16S rRNA gene sequence, or that incorporate alignment statistics.

Information from manually curated sources could also be added as a criterion or indeed as an extra filter in the first phase of data collection. One particular option is to look at the history of the strain used, and weigh sequences from strains that have seen lots of transfers between collections differently from those that have not, surmising a possible risk of contamination. Transfers can be deduced from Straininfo's Histri trees [15], a representation of the history of a strain as it moves through different biological resource centers. It is not entirely clear, though, how to weigh strains that have not been included in such a tree. Therefore, a Histri tree criterion has not yet been included in the final tests.

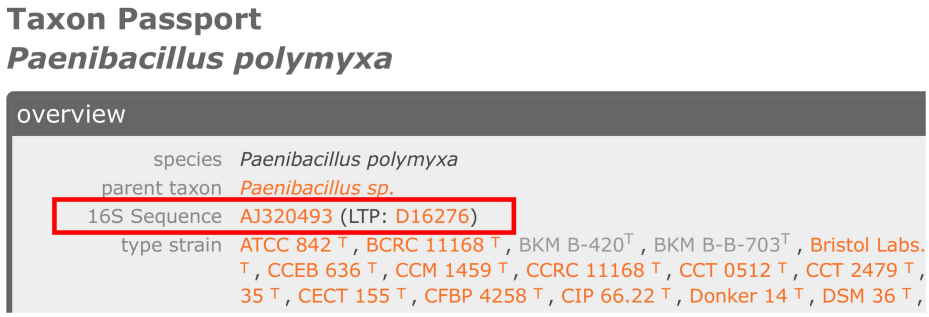


Figure 5.12: Depiction of the new taxon passport on StrainInfo, which now includes a recommendation for a high-quality 16S rRNA gene sequence, as selected by the ranking algorithm and the latest release of the LTP.

5.3.5 Display of ranking results in StrainInfo

StrainInfo displays an integrated result of strain information on a so-called strain passport, which lists equivalent strain numbers, species identification data and related data such as history, locations of strains, gene sequences and publications. Every species (subspecies) in StrainInfo also has a species (subspecies) passport collecting information on the species (subspecies), such as its type strain, and links to other related information sources. Both passports have been augmented with information about the automatically selected 16S rRNA gene sequence, as shown in Figure 5.12 for the species passport of *P. polymyxa*.

Showing a single sequence is not always straightforward, since multiple sequences sometimes end up sharing the highest rank. Currently this is resolved by displaying the accession number of the longest one. Both passports not only list the sequence selected by the automated ranking algorithm of StrainInfo, but also the sequence that has been selected in the latest release of the Living Tree Project, allowing users to compare the alternative choices.

With SeqRank, described further in Chapter 6, StrainInfo was extended to include the full ranking results, showing not only the top-ranked candidate sequence, but also all others, their criterion scores, and several options to export this information from StrainInfo. This allows users to drill deeper into the data, and provide them with a better decision support tool to use when selecting sequences, or just when examining issues with existing available 16S

rRNA gene sequences.

5.3.6 Evaluation of the automated approach

Given a set of criteria, it is instructive to see how the recommendations made by the poset ranking algorithm described here agree or disagree with those made in the Living Tree Project, which uses its own expert-dependent ranking and manual curation. For this reason, the algorithm has been run for every type strain currently described in StrainInfo that has two or more 16S rRNA gene sequences available. In order to provide a fair comparison of the ranking itself, the numbers reported here were limited to the subset of type strains for which an LTP gene sequence could be found.

From the example in Figure 5.9, it was already apparent (Table 5.1b) that the sequence selected by LTP will not be selected by StrainInfo, as it is not maximal and will occur below the top one in the final ranking. Curiously, the sequence recommended by LTP was only maximal in 2,657 (71.2% of total) type strains with more than one available sequence after filtering. When no new sequences have become available recently, this is almost always because of minor differences between the sequences by which the maximal sequences score only slightly higher on one or more attributes than the chosen LTP sequence, as in the example in this paper. In certain cases, it may also be because the algorithm used by StrainInfo has access to newer sequences (as the latest available LTP release dates from February 2013), or because an expert did not come across a particular sequence. Some issues, such as secondary structure conformity of the sequence, cannot be checked currently by the automated ranking approach, and may lead the expert to conclude that a particular 16S rRNA gene sequence really should be preferred, even if it appears of lower quality.

When the LTP sequence is maximal, it is picked as the preferred sequence by the algorithm for 1,927 type strains (72.5% of cases). The remaining choices normally include cases where an expert would choose a similar sequence that is good enough, or a better one on the basis of other implicit criteria that the ranking algorithm does not yet take into account. Often there are also several equivalent choices available, and in reality there is not one single preferred sequence, but a group of preferred sequences. This surely accounts for some of the differences in selection between the automated approach used by StrainInfo and the LTP recommendation, as choosing only a single best candidate is not always the preferred approach. In future applications, it may

well be the preferred approach to determine and show users of StrainInfo not just one highest ranking sequence, but a range of available sequences, to reflect this.

As mentioned above, the automated selection also does not have access to a high-quality alignment of the sequence, which LTP does use. This might mean that the alignment was used to remove a vector or homopolymeric stretches outside the gene in the LTP chosen sequence, while these were still erroneously included in the annotation. The automated selection algorithm would then rank the LTP chosen sequence lower than those that are properly annotated. The large number of non-maximal sequences does suggest that they might merit a closer look, and could provide a way in which to target future updates of the LTP database and improvements to the ranking algorithm.

5.4 Conclusions

In this chapter we have shown that the process of selecting from the INSDC databases a high-quality 16S rRNA gene sequence for a bacterial strain can be entirely automated. This automation is powered by the accumulative learning approach of StrainInfo that helps linking INSDC sequence records to the strains the sequences were derived from, and a poset ranking algorithm that enables ranking of candidate sequences based on multiple quality scores. This proof of concept was facilitated because we could make use of expert selections made in the All-Species Living Tree Project to construct appropriate criteria based on initial quality measures and as validation of the quality of the recommender system itself. This validation confirms the reliability of the choices made by the recommendation pipeline. The pipeline thus fits within the mission statement of StrainInfo to build an integrated platform for microbial information, where integration is completely based on automated procedures. It selects 16S rRNA gene sequences without cumbersome manual curation exhibited by LTP that is undoubtedly still superior because it takes other valuable information into account. In addition, it provides a fully curated alignment, which is outside of the scope of our approach. However, the automated approach presented here allows extending the selection of high quality 16S rRNA gene sequences beyond type strains and an up to date inclusion on a daily basis of newly reported type strain sequences which is not the case for the LTP approach. At the same time, the selection criteria can be made explicit for users to review.

As the recommendations are made available through StrainInfo, feedback of users may lead to further improvements in the annotation pipeline through additional filtering steps and new or updated quality criteria. Additionally, incorporating expert databases may be used in future to further improve filtering and implement new criteria, although it remains to be seen whether this will have a large impact on the results of the ranking algorithm. Finally, automated and manually curated approaches are a complementary means of selecting high-quality 16S rRNA gene sequences and the automated approach can in theory be extended to other housekeeping genes, which will require further work in the development of appropriate ranking criteria.

This chapter is based on the contents of the publication:
De Smet, W., De Loof, K., De Vos, P., Dawyndt, P., and De Baets, B. (2013). "Filtering and ranking techniques for automated selection of high-quality 16s rRNA gene sequences". *Systematic and Applied Microbiology*. Article accepted for publication, Sept 10, 2013.

6

Sequence Ranking in StrainInfo

6.1 Introduction

Ever since the original pioneering work by Woese *et al.* [3] in their use for phylogeny, 16S rRNA gene sequences of prokaryotic species have been widely used for strain identification purposes as well as in phylogenetic studies. Despite limitations as a phylogenetic determinant, marker sequences are still in common use today [52]. Given the central role played by the 16S rRNA gene in prokaryotic taxonomy and identification, almost all type strains have at least one 16S rRNA gene sequence that is deposited in the public INSDC database [6]. As previously established in Chapter 5, the databases often contain multiple instances of 16S rRNA gene sequences linked to different strain numbers referring to the same type or neotype strain. These sequences can all be retrieved as one group through the use of the explicit strain-culture-sequence links in StrainInfo.

In practice, not all INSDC instances of the 16S rRNA gene derived from the same strain are identical. Some of this variation occurs naturally due to minor differences between different subcultures of the same strain, or large differences due to intragenomic heterogeneity between multiple 16S rRNA cistrons [58]. Additional variation may be explained by contamination of cul-

tures, incorrect linkage between sequence records and the biological material, incomplete sequences, or sequencing errors. When relying on the 16S rRNA gene, it is important that the taxonomic information used is up-to-date, and that representative sequences used in phylogenetic studies are of the highest possible quality, since low-quality sequences might adversely impact results [66]. Furthermore, it should be possible to easily and quickly update the resulting selection of representative sequences, in response to a continually changing taxonomy and sequencing landscape. When creating or updating lists of representative 16S rRNA gene sequences is too time- or labor-consuming, researchers are reluctant to execute it frequently. This may lead to lost time due to imprecise results or even faulty conclusions made on the basis of outdated information.

Various resources already provide snapshots of current phylogenetic and taxonomy knowledge. Specialised rRNA sequence databases such as RDP [47] and SILVA [31] filter low-quality sequences and provide high-quality alignments and quality statistics to users. The All-species Living Tree Project (LTP) [60, 70] builds on the SILVA database by manually selecting and releasing a representative set of high-quality, aligned 16S rRNA gene sequences, along with a derived tree. Since this approach is based on manual curation, it is usually of very high-quality, and has become the gold standard to turn to for representative 16S rRNA gene sequences. However, in between releases, new sequences may be released and the taxonomy changed, requiring users to perform updates to selected sequences manually. Likewise, constantly increasing streams of newly released 16S rRNA gene sequences and genomes makes the manual selection process for the LTP curators increasingly time-consuming, and it is not performed for non-type strains.

As previously detailed in Chapter 5, StrainInfo contains support to automatically determine a high-quality representative 16S rRNA gene sequence for each strain in its index. SeqRank applies this algorithm to the 16S rRNA gene sequences of any strain in StrainInfo, providing users with a full list of ranked 16S rRNA gene sequences and their computed quality statistics. Using SeqRank, users can navigate to StrainInfo passport pages for every strain of interest. Performing this process for the type strains of entire genera or higher taxa in order to create a new database, update an existing one or verify choices made by the LTP is, however, still time-consuming. Therefore, a SeqRank workflow is provided that automates the entire process of fetching the current taxonomy, finding new 16S rRNA gene sequences, and extracting the most high-quality

ones. Such functionality can greatly assist any researcher in quickly rechecking phylogenetic information and availability of 16S rRNA gene sequences. It also allows them to individually scrutinize selection choices made by other researchers or by the StrainInfo ranking algorithm.

By making use of the explicit links established in StrainInfo between taxa, strains, cultures, and sequences, the SeqRank workflow applies SeqRank to the type strains of all underlying (sub)species of a taxon, starting from a taxonomic name at the genus level or above. When the type strain has been replaced by a neotype strain, the neotype strain is used. The data gathered during each stage of operation is made available to the user, under a philosophy of easy extraction of publicly available sequence data, in bulk, for further examination in appropriate analysis software. The main goal of this tool is to function as a decision support application for practicing curators and microbiologists, by allowing them to quickly build a full phylogeny of a particular taxon at the genus level or above and easily revisit choices made in the past. After selection of a technically reliable 16S rRNA gene sequence for the type strain of every (sub)species, the workflow continues to include an automatically selected outgroup sequence in a multiple alignment and phylogenetic tree based on selected 16S rRNA gene sequences.

6.2 SeqRank workflow

SeqRank can be launched from any strain passport in StrainInfo. It generates an overview of all 16S rRNA gene sequences of the selected strain and the attributes of those strains used to rank them, as previously discussed in Chapter 5. The SeqRank workflow, which can be launched from any taxon passport at genus level or above, is restricted to 700 type strains (the approximate size of the biggest family in StrainInfo at time of writing), in order to prevent client and server overload for overly large taxonomies. Its main steps are shown in Figure 6.1. These steps correspond to the manual lookup process a researcher would follow. However, the SeqRank workflow is completely automated, allowing manual intervention whenever applicable. It has been divided in two main subworkflows: the collection of high-quality 16S rRNA gene sequences for the type strains of all (sub)species found in the given taxon, and the creation of a simple phylogenetic tree for visualisation purposes.

When performing the ranking process starting from a given taxon, the log-

ical first step is to ascertain the current taxonomic status of all (sub)species underlying the taxon, using recent publications or species lists from external resources. This is the first step in the workflow ("subtaxa query") which will search all (sub)species in the taxonomic subtree rooted at the given taxon for all currently valid species names. Every species has a type strain, which is the name bearer of the species, and the source of 16S rRNA gene sequences we are interested in. As the type strain may (and usually does) have multiple subcultures available, assigned different strain numbers by different researchers and culture collections, a thorough search of sequence records must be done by searching for all sequence records that refer to one of the possible equivalent strain numbers of the type strain. The second step in the workflow queries StrainInfo for all equivalent strain numbers of the type strain. This information is successively used to retrieve all sequence records that state in their annotations or descriptions that they were derived from the type strain (step 3, "16S rRNA gene sequence collection"), after which all that remains is filtering out bad sequences and deciding which one to use as a representative for each type strain. This is the last step ("best 16S sequence calculation"), which when done in SeqRank uses the algorithms described previously in Chapter 5.

Each step can be modelled as a computational process in which certain input data are given, data collection or data processing is performed, and an output is generated which forms the input for the next step of the workflow. Data collection is based on data integrated from a variety of external data resources, available in StrainInfo (see also Chapter 2).

After generation of the list of 16S rRNA gene sequences, a tree is also constructed, using the secondary workflow on the right in Figure 6.1. This phylogenetic tree is not meant to be a publication-ready tree, but provides users with an intuitive visualisation of the current selection of sequences. For taxa where the 16S rRNA gene is informative enough as a phylogenetic marker, it can also be used as a way to detect issues with selected sequences. Aberrant sequences will tend to distort the phylogenetic tree, or to an expert eye cluster in unexpected ways. When no such aberrant sequences are included, the tree is also suitable as a quick way to examine the phylogenetic makeup of a particular taxon. The SeqRank workflow thus produces two main output results for a given taxon: a list of high-quality 16S rRNA gene sequences, and a phylogenetic tree. But, as shown below, data output is made available to the user at any stage, which allows users to perform their own analysis of all underlying data using specialised tools, and frees the application to focus on its core tasks of data

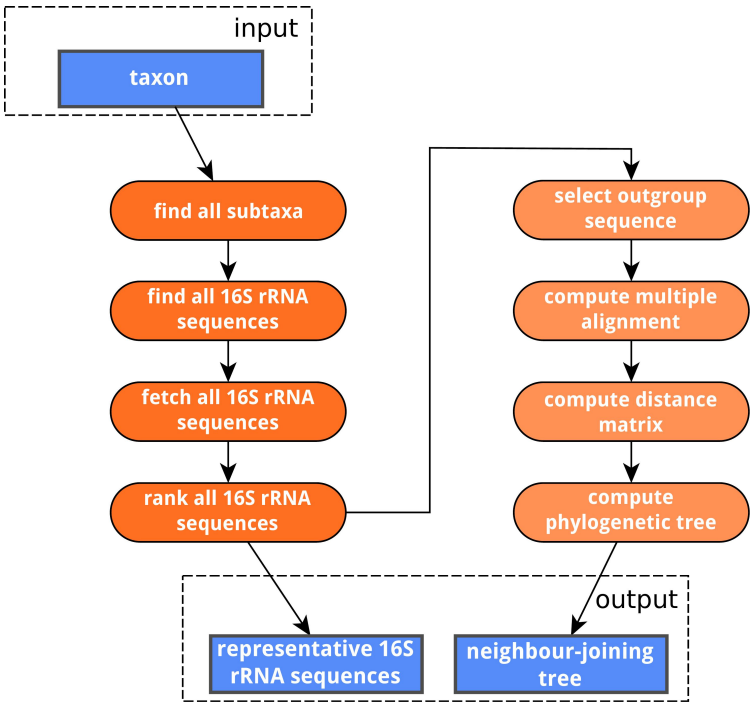


Figure 6.1: SeqRank workflow followed during sequence selection and subsequent phylogenetic tree generation. The input is any taxon in the Bacteria or Archaea at the level of genus or above. A series of queries fetches all (sub)species underlying the given taxon, type strains for each (sub)species, and 16S rRNA gene sequences for each type strain. Finally, a poset ranking algorithm is applied to select a high-quality 16S rRNA gene sequence for each type strain. After selection, an example phylogenetic tree is generated by automatically selecting an outgroup sequence, computing a multiple alignment and distance matrix and constructing a neighbor-joining tree.

collection and quality ranking of 16S rRNA gene sequences.

6.3 Detailed workflow

The SeqRank workflow is integrated into the StrainInfo platform, as an online application, accessible from any taxon passport at the genus level or higher. Figure 6.2 displays part of the taxon passport for the genus *Stenotrophomonas*, where a link to the SeqRank workflow is highlighted. Clicking this link opens the application, which automatically starts the data collection process.

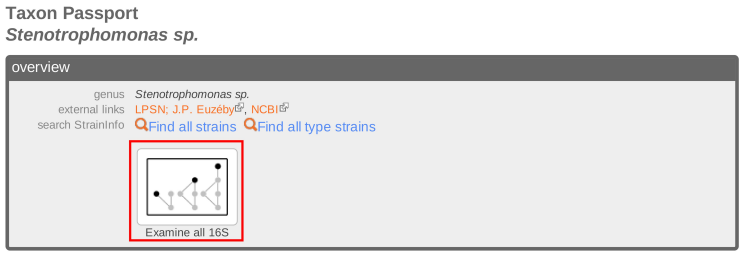


Figure 6.2: Screenshot of the overview panel on the taxon passport for the genus *Stenotrophomonas*[1]. The SeqRank workflow is accessed by following the highlighted link, which appears on taxon passports at the genus level or above.

Figure 6.3 shows the end result after the whole workflow has been executed. It is modeled to resemble the workflow shown in Figure 6.1, providing the user with an overview of all the necessary steps. Each task is presented in Figure 6.3 by a separate block, which fills up as the workflow progresses past it. The input of each task corresponds to the output of the previous task. The first task is a special case, as its input is the taxon name that the user selected by opening the application from a certain taxon passport. In light of its goal to allow users to extract data and move on from any point with their own analysis, each SeqRank workflow task can be selected to reveal its resulting output. Therefore the SeqRank workflow results can be exported or further investigated by following provided links. Since runs can be somewhat lengthy (especially when a phylogenetic tree is built dynamically), the output of any currently completed task can always be examined, without interrupting calculation.

[1] `http://www.straininfo.net/taxa/2610`

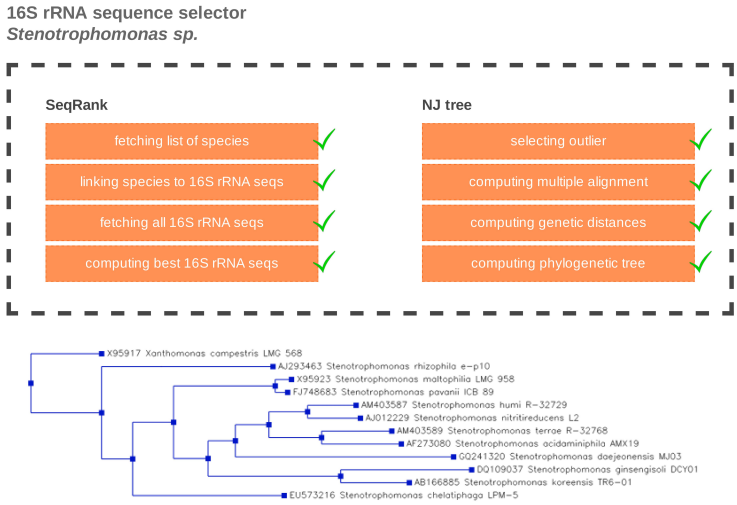


Figure 6.3: Display of the SeqRank application after completion. The last step of the workflow (computed phylogenetic tree) is selected in this figure.

6.3.1 Finding (sub)species

SeqRank, as accessed from the strain passport, selects a representative 16S rRNA gene sequence for a single strain. The SeqRank workflow enables users to repeat this process for a larger collection of type strains, by performing the selection for all (sub)species underlying a certain input taxon, which can be any taxon at the rank of genus or above. This leverages the taxonomic tree to conveniently select a collection of strains, without having to manually select each strain separately. While each species in this taxonomic tree represents a set of strains with common features, defined in terms of their phenotype or in terms of genomic similarity [24], the type strain of that species acts as the name bearer and representative of the species. Representative 16S rRNA gene sequences for each species must hence be derived from its type strain. In the first step of the workflow, the names of all species or subspecies that occur in the taxonomic tree under the taxon are ascertained. These names are used in subsequent steps to determine the strain numbers of their respective type strains, with which sequence records of these strains can be found.

The delineation of strains in various taxonomic groups is however based on a scientific consensus that is constantly in flux. At least since the 1970s, the

Bacteriological Code has contained clear instructions on formal publication and registration of taxonomic changes [8], introducing the concept of a *validly* published name. In order to be validly published, the name must appear in the International Journal of Systematic Bacteriology, later renamed as the International Journal of Systematic and Environmental Microbiology (IJSEM). The rules ensure a permanent record in the publication record of all accepted species names, their appointed type strains and, in recent years, associated 16S rRNA gene sequences sequenced in the context of new species descriptions. While disagreements in taxonomic nomenclature and taxon delineation do occur, the publication history of validly accepted names form a consensus taxonomy that can be used to lookup (sub)species names in the first step of the workflow.

Unfortunately, no standard electronic representation of the taxonomic tree and previous taxonomic changes was ever adopted. Taxonomic information in StrainInfo must therefore be determined from third-party taxonomic databases, who produce lists of accepted taxonomic names, their relations and how this has changed over time. The most complete and up-to-date resources are the Bacterial Nomenclature Up-to-Date, available from the Deutsche Sammlung von Mikroorganismen und Zellkulturen (DSMZ)[2], and the List of Prokaryotic Names with Standing in Nomenclature [11] (LPSN). Both resources are quite complete, but do not generally contain the same complementary information. Since LPSN contains more strain numbers and a better description of taxonomic changes, it is used as the preferential source for taxonomic tree information. LPSN is available online as a set of hand-crafted HTML web pages, which are amenable to automatic parsing, although fragile screen scraping must be used. More details on the implementation of this integration can be found in Chapter 3.

The taxonomic information from LPSN is used to perform selection of all (sub)species underlying the taxon of interest. When the taxon of interest is a species, this task is usually trivial, although the species might still be broken up into several subspecies. For taxa at the rank of genus or above, the entire taxonomic tree underlying the input taxon is searched for validly accepted names of species or subspecies. This may include names that were at one point validly accepted, but have since been the target of a taxonomic change,

[2] `http://www.dsmz.de/bacterial-diversity/prokaryotic-nomenclature-up-to-date.html`

Figure 6.4: Popup shown when the user moves the cursor over the basonym indicator (a red square) next to a taxonomic name in the SeqRank workflow. The listed name is the last validly published synonym of the basonym known to StrainInfo.

which usually means that the species has been reassigned to a different genus, or that it has been deemed a synonym of a different name, with the other name taking preference. In these cases, the older name is referred to as the basonym. The basonym remains a valid name, but any work which bases itself on the current status of the taxonomic tree should refer to the species by its newer name. This also means that when the input taxon is, for example, a genus, users will in general prefer to exclude type strains of these basonyms from consideration. For instance, the species *dokdonensis* originally belonged to the genus *Stenotrophomonas* [86], but was later transferred to the genus *Pseudoxanthomonas* [87], and so it should not appear in phylogenetic trees of the genus *Stenotrophomonas*. By using the transfer history extracted from LPSN, the basonym can be included in all species lists, but is by default not selected to be included in the list of representative 16S rRNA gene sequences or the phylogenetic tree created by the SeqRank workflow. In all steps of the SeqRank workflow, a basonym also has a red mark next to its name. On mouse-over, a popup displays the last accepted name under which the species is known. The effect is illustrated in Figure 6.4. Since all species names are linked to their taxon passport, this provides a natural way to explore changes in genera or families.

6.3.2 Linking type strains

When describing a new species, the bacteriological code specifically requires the appointment of a type strain that must be submitted to at least two different public culture collections [8]. In theory, it should thus be possible to determine the strain numbers assigned to type strains through text mining of the original publication. However, while this option is worth considering, type

strain information is also extracted from publications by the culture collections themselves, and by LPSN, where it likely can be extracted with greater accuracy. StrainInfo therefore uses LPSN as the authoritative source of type strain information. Strain numbers included in LPSN are processed by StrainInfo's strain integration workflow, originally described in [10], which allows StrainInfo to update strain information and explicitly link the taxonomic names found on the LPSN website to the strain numbers of their type strain. Updates of type strain information are performed weekly, as part of the StrainInfo database update pipeline (Chapter 3).

As pointed out in Chapter 5, 16S rRNA gene sequences are extracted from all publicly available sequence records in the prokaryotes section of the European Nucleotide Archive (ENA) [41], one of the databases forming the International Sequence Database Collaboration (INSDC) [6]. There are various ways in which (type) strain information is then linked to molecular sequences. Annotation and quality problems, some of which are well documented [84], preclude using the species identification of these sequences directly. However, sequence records listed in any of the INSDC databases routinely identify the source strain through its strain number in their metadata. StrainInfo links a molecular sequence to a certain culture of a strain if the strain number mentioned in the record can be resolved to that particular culture, either because the strain number uniquely refers to a specific culture, or because the combination of strain number and species name (also retrieved from the sequence record) is unique. Secondary data sources augment this information. In particular, the data from LPSN often contains a list of 16S rRNA gene sequence records of the type strain of a (sub)species. This information is also linked to the type strain during processing of LPSN.

A last possible way to link 16S rRNA gene sequences to type strains is to use periodically released phylogenetic tree of Bacteria and Archaea from the Living Tree Project, which uses a manually selected 16S rRNA gene sequence for each included species. As discussed in Chapter 5, about 1% of the sequences used in this tree were not previously linked to the type strain in StrainInfo. The discussion in that chapter also showed, however, that these 16S rRNA gene sequences cannot simply be linked to the type strain, as a substantial portion of them were shown to either be mistakenly chosen or not verifiably derived from the type strain. Therefore, manual verification is always needed when using the LTP data set in such a manner.

Putting all of the above techniques together, a set of sequence records can

Species	Strains	16S	Location
Stenotrophomonas acidaminiphila	AMX19^T	AF273080	<1..>1542
Stenotrophomonas africana[a]	CCUG 41684^T	GU945534	<1..>1427
	MGB^T	U62646	1..1452
Stenotrophomonas chelatiphaga	LPM-5^T	EU573216	<1..>1454
Stenotrophomonas daejeonensis	MJ03^T	GQ241320	<1..>1420
Stenotrophomonas dokdonensis[b]	DS-16^T	DQ178977	<1..>1500
Stenotrophomonas ginsengisoli	DCY01^T	DQ109037	<1..>1467
Stenotrophomonas humi	R-32729^T	AM403587	<1..>1494
Stenotrophomonas koreensis	TR6-01^T	AB166885	<1..>1463
Stenotrophomonas maltophilia	ATCC 13637^T	AB008509	<1..>1467
		FJ971878	<1..>797
		M59158	1..1545
	CGMCC 1.1788^T	DQ067559	<1..>1412
	IAM 12423^T	AB294553	<1..>1538
	LMG 958^T	X95923	1..1500
Stenotrophomonas nitritireducens	L2^T	AJ012229	1..1513
Stenotrophomonas pavanii	ICB 89^T	FJ748683	<1..>1483
	LMG 25348^T	HQ641452	<1..>1497
Stenotrophomonas rhizophila	e-p10^T	AJ293463	7..>1537
Stenotrophomonas terrae	R-32768^T	AM403589	<1..>1512

[a] Heterotypic synonym of *Stenotrophomonas maltophilia*

[b] Homotypic synonym of *Pseudoxanthomonas dokdonensis*

Table 6.1: Example of the table returned after lookup of all the type strains of all (sub)species in the genus *Stenotrophomonas* and extraction of all 16S rRNA gene sequences of the type strains.

be retrieved for the type strain of any species in StrainInfo. This list is then narrowed down to only 16S rRNA gene sequences using feature annotations or description and title fields (when no annotations are available). The result is a list of type strains and their associated 16S rRNA gene sequences, as exemplified in Table 6.1 for the genus *Stenotrophomonas*. In it, retrieved 16S rRNA gene sequences are shown with the strain number of the culture they are derived from, and the species the strain represents. Species names, strain numbers and accession numbers of sequence records are all linked to their respective passport pages in StrainInfo, which makes it easy to find more detailed information, for instance by navigating to the Biological Resource Center (BRC) catalog pages, that may contain additional information on the strain. The last column of the table lists the location of the 16S rRNA gene sequence within the raw sequence record. This allows to discern the difference between multiple 16S rRNA gene sequences extracted from the same sequence record, as might be the case with complete or draft genome sequence records.

6.3.3 Extraction of 16S rRNA gene sequences

After having located all 16S rRNA genes, the sequences can be downloaded after they have been extracted from the sequence records (third task). This task uses the accession numbers and location field of feature annotations (shown in the last two columns of Table 6.1) to extract 16S rRNA gene sequence from INSDC sequence records. The resulting gene sequences are formatted in FASTA format and available for direct copy or download.

The FASTA format was originally the input and output format of a line of sequence alignment search programs. This line started with FASTP, a protein-protein local similarity search program [88], that was later improved upon and extended to work on nucleotide sequences, as the FASTA [89] program (the 'A' in FASTA stands for 'All', as opposed to the 'P' of 'Protein'). Figure 6.5 depicts an example sequence in FASTA format. A sequence record in a FASTA file is nothing more than a header line (starting with >), which contains a general description and a number of lines containing the sequence using the chemical letter codes for the nitrogen bases of the corresponding nucleotides standardized by the International Union of Pure and Applied Chemistry (IUPAC) [90]. The possible codes are those for the pyrimidine bases *Cytosine* (C), *Thymine* (T) and *Uracil* (U), and the purine bases *Guanine* (G) and *Adenine* (A). As sequencing technology is not perfect, the identity of the base at a certain position

```
>X95923.2|1..1500|X95923 Stenotrophomonas maltophilia LMG 958
agtgaacgctggcggtaggcctaacacatgcaagtcgaacggcagcacaggagagcttgctctctgggtggcgagtggcg
gacgggtgaggaatacatcggaatctacttttttcgtgggggataacgtagggaaacttacgctaataccgcatacgacct
acgggtgaaagcaggggatcttcggaccttgcgcgattgaatgagccgatgtcggattagctagttggcggggtaaaggc
ccaccaaggcgacgatccgtagctggtctgagaggatgatcagccacactggaactgagacacggtccagactcctacgg
gaggcagcagtggggaatattggacaatgggcgcaagcctgatccagccataccgcgtgggtgaagaaggccttcgggtt
gtaaagccctttgttgggaaagaaatccagctggttaatacccggttgggatgacggtacccaaagaataagcaccggc
taacttcgtgccagcagccgcggtaatacgaagggtgcaagcgttactcggaattactgggcgtaaagcgtgcgtaggtg
gtcgtttaagtccgttgtgaaagccctgggctcaacctgggaactgcagtggatactggrcgactagagtgtggtagagg
gtagcggaattcctggtgtagcagtgaaatgcgtagagatcaggaggaacatccatggcgaaggcagctacctggaccaa
cactgacactgaggcacgaaagcgtggggagcaaacaggattagataccctggtagtccacgccctaaacgatgcgaact
ggatgttgggtgcaatttggcacgcagtatcgaagctaacgcgttaagttcgccgcctggggagtacggtcgcaagactg
aaactcaaaggaattgacgggggcccgcacaagcggtggagtatgtggtttaattcgatgcaacgcgaagaaccttacct
ggccttgacatgtcgagaactttccagagatggattggtgccttcgggaactcgaacacaggtgctgcatggctgtcgtc
agctcgtgtcgtgagatgttgggttaagtcccgcaacgagcgcaacccttgtccttagttgccagcacgtaatggtggga
actctaaggagaccgccggtgacaaaccggaggaaggtggggatgacgtcaagtcatcatggcccttacggccagggcta
cacacgtactacaatggtagggacagagggctgcaagccggcgacggtaagccaatcccagaaaccctatctcagtccgg
attggagtctgcaactcgactccatgaagtcggaatcgctagtaatcgcagatcagcattgctgcggtgaatacgttccc
gggccttgtacacaccgcccgtcacaccatgggagtttgttgcaccagaagcaggtagcttaaccttcgggagggcgctt
gccacggtgtggccgatgactggggtgaagtcgtaacaaggtagccgtatcggaaggtgc
```

Figure 6.5: Example gene sequence record in FASTA format. The header of the record includes its accession number and version, the location from which the sequence was extracted in the original INSDC sequence record, and the description field as extracted from the INSDC sequence record.

might be ambiguous. The presence of such an ambiguous base (or base pair) is generally indicated with the letter 'N', or a selection of more rarely used codes which correspond to any combination of the four bases that occur in DNA (Cytosine, Thymine, Guanine and Adenine)[3]. Sequences are normally broken up into multiple lines, each 80 characters long, so tools can use a fixed size buffer to process them. This practice is also followed by the SeqRank workflow. Multiple sequences can be downloaded as a multi-FASTA file, which is simply a concatenation of several single FASTA records, where the header line indicates the start of each sequence record. While BLAST (Basic Local Alignment Search Tool) offers similar sensitivity at greater speed [91] and has largely replaced FASTA as the similarity search tool of choice, support for the FASTA file format remains standard in most molecular sequence analysis tools. This can be attributed to its ease of parsing and functional simplicity. Providing the sequence listings in FASTA format thus ensures maximum compatibility with a wide variety of existing sequence analysis tools.

[3] A listing of these codes can be found at `http://www.ncbi.nlm.nih.gov/staff/tao/tools/tool_lettercode.html`

6.3.4 High-quality sequence selection

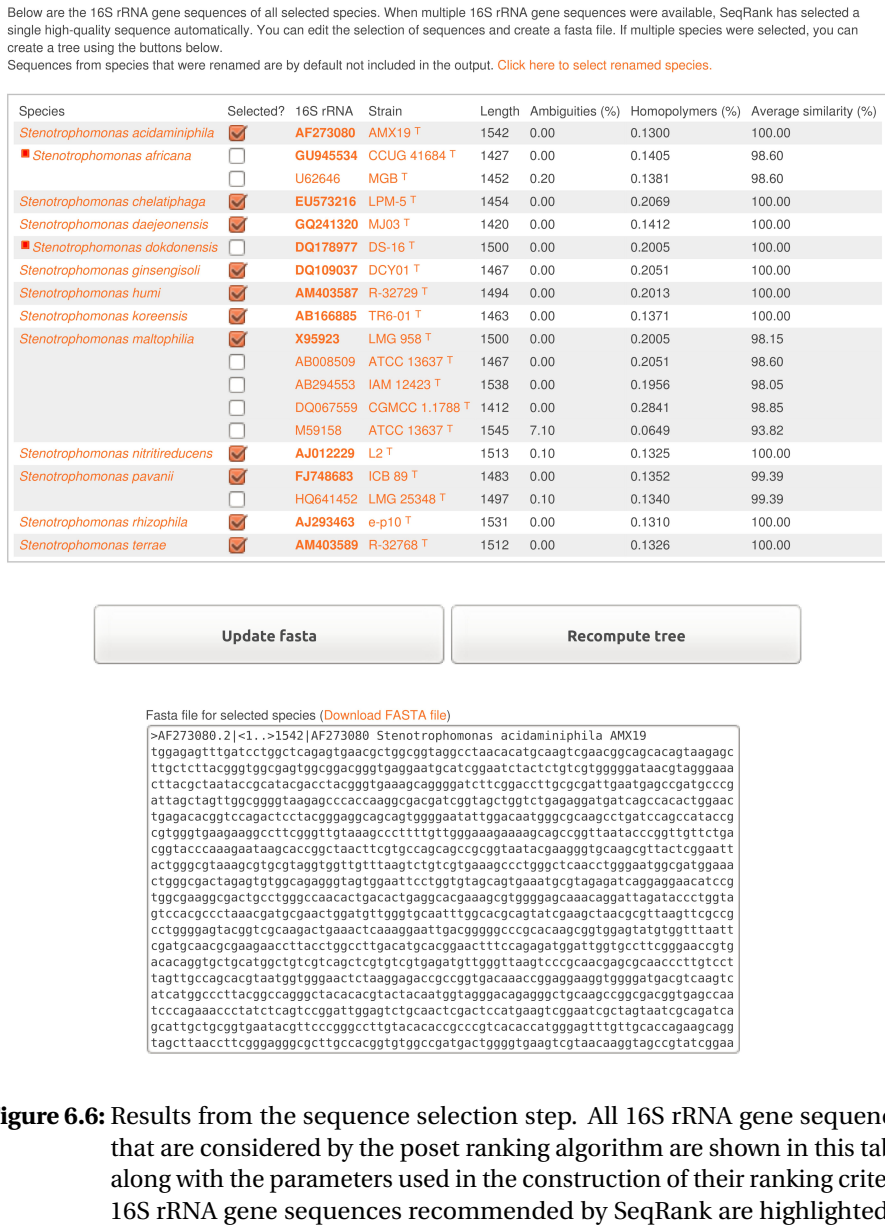
The complete list of 16S rRNA gene sequences collected in the previous step can be useful for a general analysis. However, it still includes low-quality sequences that should be filtered out. It is also commonly the case that studies include only a single representative 16S rRNA gene sequence per type strain. The next step of the SeqRank workflow therefore performs filtering of the list of sequences, and provides ways to examine the results. As such, an automated algorithm can greatly reduce the time needed to select a 'good' set of sequences, while still providing the option to override those automatic choices where expert knowledge leads to other preferences.

Selection of a recommended 16S rRNA gene sequence and the SeqRank algorithm were first introduced in Chapter 5, which describes the partially ordered set (poset) ranking algorithm used in StrainInfo. As the reader might recall, this algorithm is based on building a mathematical construct, the poset, that compares sequences based on four criteria. These criteria are preference functions that use four different sequence attributes as their input. These attributes are:

Sequence length Length is normally measured in base pairs. In general, longer sequences provide more information for multiple alignment and phylogenetic tree building algorithms to work with. On the other hand, over-long sequences might be a sign of annotation error or some other issue with the sequence submission. Therefore, this value should be somewhere near the typical size of a 16S rRNA gene sequence, values of which are taken into account into the criteria calculation.

Ambiguity percentage As introduced in the previous section, molecular sequences are often not perfectly read, and contain indicators of so-called ambiguities: bases in the sequenced strand of which the identity is unknown. This lack of information can cause issues for phylogenetic tree building or any sort of clustering approach used in identification, and the percentage of such ambiguities in the entire sequence should therefore be as low as possible.

Homopolymer percentage 16S rRNA gene sequences typically only contain a small percentage of stretches of the same base, called homopolymeric stretches. The presence of more than the average of such stretches in a



Below are the 16S rRNA gene sequences of all selected species. When multiple 16S rRNA gene sequences were available, SeqRank has selected a single high-quality sequence automatically. You can edit the selection of sequences and create a fasta file. If multiple species were selected, you can create a tree using the buttons below.
Sequences from species that were renamed are by default not included in the output. Click here to select renamed species.

Species	Selected?	16S rRNA	Strain	Length	Ambiguities (%)	Homopolymers (%)	Average similarity (%)
Stenotrophomonas acidaminiphila	☑	**AF273080**	AMX19 T	1542	0.00	0.1300	100.00
Stenotrophomonas africana	☐	**GU945534**	CCUG 41684 T	1427	0.00	0.1405	98.60
	☐	U62646	MGB T	1452	0.20	0.1381	98.60
Stenotrophomonas chelatiphaga	☑	**EU573216**	LPM-5 T	1454	0.00	0.2069	100.00
Stenotrophomonas daejeonensis	☑	**GQ241320**	MJ03 T	1420	0.00	0.1412	100.00
Stenotrophomonas dokdonensis	☐	**DQ178977**	DS-16 T	1500	0.00	0.2005	100.00
Stenotrophomonas ginsengisoli	☑	**DQ109037**	DCY01 T	1467	0.00	0.2051	100.00
Stenotrophomonas humi	☑	**AM403587**	R-32729 T	1494	0.00	0.2013	100.00
Stenotrophomonas koreensis	☑	**AB166885**	TR6-01 T	1463	0.00	0.1371	100.00
Stenotrophomonas maltophilia	☑	**X95923**	LMG 958 T	1500	0.00	0.2005	98.15
	☐	AB008509	ATCC 13637 T	1467	0.00	0.2051	98.60
	☐	AB294553	IAM 12423 T	1538	0.00	0.1956	98.05
	☐	DQ067559	CGMCC 1.1788 T	1412	0.00	0.2841	98.85
	☐	M59158	ATCC 13637 T	1545	7.10	0.0649	93.82
Stenotrophomonas nitritireducens	☑	**AJ012229**	L2 T	1513	0.10	0.1325	100.00
Stenotrophomonas pavanii	☑	**FJ748683**	ICB 89 T	1483	0.00	0.1352	99.39
	☐	HQ641452	LMG 25348 T	1497	0.10	0.1340	99.39
Stenotrophomonas rhizophila	☑	**AJ293463**	e-p10 T	1531	0.00	0.1310	100.00
Stenotrophomonas terrae	☑	**AM403589**	R-32768 T	1512	0.00	0.1326	100.00

Update fasta | Recompute tree

Fasta file for selected species (Download FASTA file)

```
>AF273080.2|<1..>1542|AF273080 Stenotrophomonas acidaminiphila AMX19
tggagagtttgatcctggctcagagtgaacgctggcggtaggcctaacacatgcaagtcgaacggcagcacagtaagagc
ttgctcttacgggtggcgagtggcggacgggtgaggaatgcatcggaatctactctgtcgtgggggataacgtagggaaa
cttacgctaataccgcatacgacctacgggtgaaagcaggggatcttcggaccttgcgcgattgaatgagccgatgcccg
attagctagttggcggggtaaggcccaccaaggcgacgatcggtagctggtctgagaggatgatcagccacactggaac
tgagacacggtccagactcctacgggaggcagcagtggggaatattggacaatgggcgcaagcctgatccagccataccg
cgtgggtgaagaaggccttcgggttgtaaagccctttgttgggaaagaaaagcagccggttaataacccggttgttctga
cggtacccaaagaataagcaccggctaacttcgtgccagcagccgcggtaatacgaagggtgcaagcgttactcggaatt
actgggcgtaaagcgtgcgtaggtggttgtttaagtctgtcgtgaaagccctgggctcaacctgggaatggcgatggaaa
ctgggcgactagagtgtggcagagggtagtggaattcctggtgtagcagtgaaatgcgtagagatcaggaggaacatccg
tggcgaaggcgactgcctgggccaacactgacactgaggcacgaaagcgtggggagcaaacaggattagataccctggta
gtccacgccctaaacgatgcgaactggatgttgggtgcaatttggcacgcagtatcgaagctaacgcgttaagttcgccg
cctggggagtacggtcgcaagactgaaactcaaaggaattgacgggggcccgcacaagcggtggagtatgtggtttaatt
cgatgcaacgcgaagaaccttacctggccttgacatgcacggaactttccagagatggattggtgccttcgggaaccgtg
acacaggtgctgcatggctgtcgtcagctcgtgtcgtgagatgttgggttaagtcccgcaacgagcgcaacccttgtcct
tagttgccagcacgtaatggtgggaactctaaggagaccgccggtgacaaaccggaggaaggtggggatgacgtcaagtc
atcatggcccttacggccagggctacacacgtactacaatggtagggacagagggctgcaagccggcgacggtgagccaa
tcccagaaaccctatctcagtccggattggagtctgcaactcgactccatgaagtcggaatcgctagtaatcgcagatca
gcattgctgcggtgaatacgttcccgggccttgtacacaccgcccgtcacaccatgggagtttgttgcaccagaagcagg
tagcttaaccttcgggagggcgcttgccacggtgtggccgatgactggggtgaagtcgtaacaaggtagccgtatcggaa
```

Figure 6.6: Results from the sequence selection step. All 16S rRNA gene sequences that are considered by the poset ranking algorithm are shown in this table, along with the parameters used in the construction of their ranking criteria. 16S rRNA gene sequences recommended by SeqRank are highlighted in bold face. All recommended 16S rRNA gene sequences that do not belong to a basonym are automatically preselected for inclusion in the FASTA file returned to users and the phylogenetic tree that is built in the next stage of the workflow.

16S rRNA gene sequence might then indicate a problem with sequencing. For instance, one of the most well-known failure modes of pyrosequencing, a next-generation sequencing technique, is to overestimate the length of a homopolymer [92].

Average similarity Multiple 16S rRNA gene sequences are often available for the same strain. With the exclusion of sequences from strains that show strong internal variability between 16S rRNA cistrons, the 16S rRNA gene sequences from different subcultures of the same type strain are expected to be highly similar. Contamination or annotation errors may however result in the inclusion of 16S rRNA gene sequences of different strains in the set of available 16S rRNA gene sequences. A measure of the average similarity of available sequences to each other is used to detect outliers.

The average similarity measure is based on the sequence distance $d(S_i, S_j)$, which is the ratio of the number of matching base pairs in the pairwise alignment of sequence S_i and S_j, to the length of the shortest of both sequences. The average similarity $av(S_i)$ of a sequence S_i from a set of sequences $\{S_1, S_2, \ldots, S_n\}$ of the same type strain is then:

$$av(S_i) = \frac{\sum_{j=1, j\neq i}^{n} d(S_i, S_j)}{n-1}$$

Even with limited background knowledge on the gene, this information is still highly intuitive to parse for most users, since it corresponds directly with known physical attributes of the sequence, of which optimal values (a length close to 1542 bp for instance) are well known. Criteria values, on the other hand, are values between zero and one, where larger is better. Their values are much harder to connect directly to the physical properties of the sequence. Therefore, the SeqRank workflow only shows the physical attributes in the results table (Figure 6.6). This allows users to identify poor quality sequences without the need of any further analysis, and it shows them how the algorithm may have decided that a sequence was of higher quality.

As shown in Figure 6.6, the result list of this last main SeqRank task also includes toggle switches. The output FASTA file from SeqRank can be altered by toggling these switches, for instance allowing users to remove sequences of certain type strains entirely, including certain or all renamed species, or select

multiple sequences of the same type strain. Additionally, using a form control at the bottom of the table, the tree building process can be restarted with this new selection of sequences. In this way, different combinations of sequences can be selected and visualised.

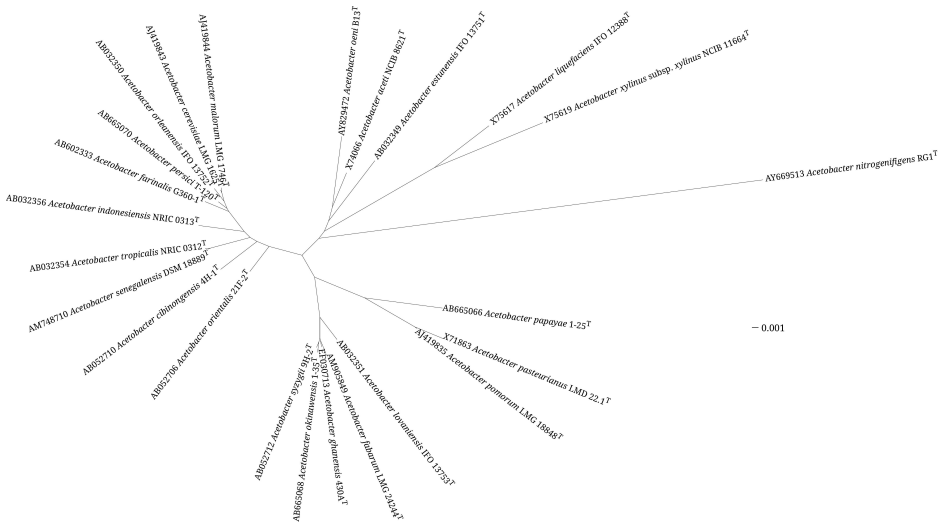
6.3.5 Tree building

Constructing publication-ready phylogenetic trees often requires some manual intervention, such as in adjustments of automated alignment artifacts [93]. The tree provided by the SeqRank workflow, however, is merely meant to provide a visualisation of the relationship between a reasonably closely related set of sequences, and needs to deliver this visualisation in an acceptably short running time. Since some of the more accurate tree building algorithms are very time consuming, a trade-off must be made here between accuracy and speed. The methods described below therefore rely mostly on computationally inexpensive approaches for phylogenetic tree building.

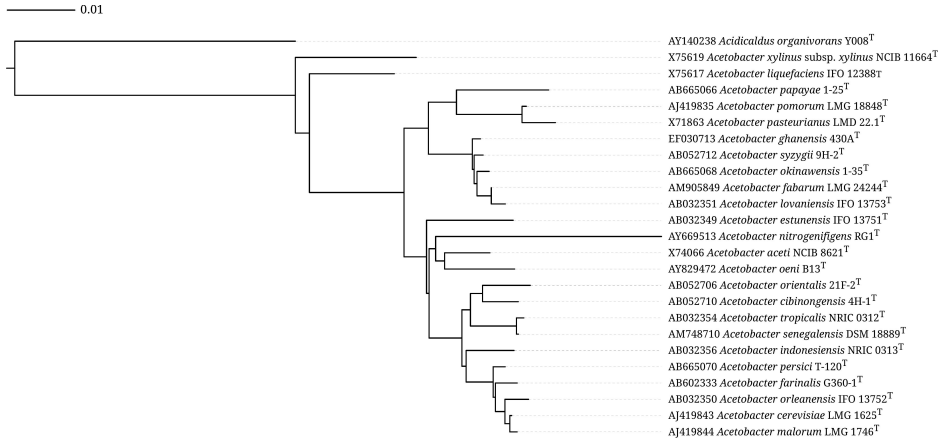
Outgroup selection

In order to make the tree visualisation easier to interpret, the SeqRank workflow produces a phylogenetic tree including all selected 16S rRNA gene sequences. Figure 6.7 illustrates the difference between a rooted and unrooted tree, using two trees based on SeqRank selected sequences and drawn using the Interactive Tree of Life [94, 95]. An unrooted tree, such as the one in Figure 6.7a, visualises relative evolutionary difference of all species in the tree, just as the rooted tree, but in a rooted tree the choice of root corresponds with a hypothetical common ancestor of all strains. A rooted tree, such as in Figure 6.7b, is often easier to read, as the order of internal nodes can be treated as significant by the reader, provided the root was placed correctly.

In order to provide a rooted tree, an outgroup sequence must be chosen. An outgroup is a species which is known or thought to have branched off the phylogenetic tree before all other taxa in the tree. The point where this outgroup branched off can then be surmised to be the root of the tree. A good outgroup should thus be distantly related to all taxa in the tree. Using sequences of outgroups that are too distantly related to the other taxa in the tree, however, will lead to the relationship of closely related taxa being obscured. A good outgroup should therefore be known to have branched off previously,



(a) Unrooted phylogenetic tree of genus *Acetobacter*



(b) Rooted phylogenetic tree of genus *Acetobacter*

Figure 6.7: Two phylogenetic trees of 16S rRNA gene sequences recommended by the SeqRank workflow, for all type strains of the genus *Acetobacter*. Both trees were generated from the same set of sequences, except for the outgroup *Acidicalcus organivorans*. Figure 6.7a is an unrooted tree, displaying relative evolutionary distance between species. Internal nodes indicate which species are more closely related than others. Figure 6.7b is rooted by using the outgroup included by using the SeqRank workflow.

but not be so distantly related that it will reduce the information content of the multiple alignment used in tree reconstruction. The root of the tree can then be placed at the internal node where the outgroup is connected to the rest of the phylogenetic tree.

Choosing an outgroup sequence is the first step of the tree building process. It is based on a simple heuristic: closely related strains are likely (but not necessarily) found in sibling taxa of the species, genus or family used as input. Finding these siblings of course requires access to a taxonomy at the family level and higher. In, SeqRank, the taxonomy used is again that of LPSN, which especially at the higher order hews closely to the taxonomy of Bergey's Manual of Systematic Bacteriology[96, 97], but also incorporates information from various information sources (further information can be found at the LPSN website[4]). For the purpose of this discussion, this taxonomy can be thought of as a tree, where the leafs are subspecies and species, and each taxon has a parent at a higher level, up until the level of domain, at which are found the three domains: Bacteria, Archaea, and Eukarya.

Outgroup selection is performed according to Algorithm 1. It presupposes the existence of a function "parent", which returns the parent of a taxon in its taxonomic tree, and a function "type_taxon". This latter function returns the subtaxon of a given taxon that is considered to be the 'type taxon' of its parent. For instance, every genus has a type species, every family has a type genus, and so on... The type strain of a given higher order taxon is found by following the chain of type taxa down to the species level and returning its type strain. Algorithm 1 finds a suitable outgroup by selecting a type strain from the siblings of the input taxon if possible. First, it will attempt to use the sibling that is the type taxon of the common ancestor of the siblings of the input taxon. If this fails, the alphabetically first non-type sibling is checked, which can be necessary, for example, if no type strain is available for the type taxon, or the input taxon *is* the type taxon. The procedure is repeated at a higher level, until a type strain is found.

Tree reconstruction

Phylogenetic tree reconstruction algorithms are most often divided into two groups: distance-based methods and character-based methods. Distance-

[4] `http://www.bacterio.cict.fr/classification.html`

Input: Taxon t
Output: 16S rRNA gene sequence of the type strain of an outgroup taxon
$parent \leftarrow parent(t)$;
while *parent has no type taxon or type taxon has no type strain* **do**
    $parent \leftarrow parent(parent)$;
$candidate\_strain \leftarrow type\_strain(type\_taxon(parent))$;
if $candidate\_strain$ *exists and* $candidate\_strain \neq type\_strain(t)$
then
    return $candidate\_strain$;
else
    $parent \leftarrow parent(t)$;
    while $parent$ *exists* **do**
        foreach *subtaxon of parent in alphabetic order* **do**
            $candidate\_strain \leftarrow type\_strain(subtaxon)$;
            if $candidate\_strain$ *exists and*
            $candidate\_strain \neq type\_strain(t)$ **then**
                return $candidate\_strain$;
        $parent \leftarrow parent(parent)$

Algorithm 1: Outgroup selection algorithm

based methods rely on a measure of evolutionary distance, captured in a distance matrix almost always derived from the multiple sequence alignment of the selected sequences. Character-based methods, on the other hand, attempt to reconstruct the evolutionary history directly from the data in the multiple sequence alignment [98]. The former group includes traditional clustering approaches such as Unweighted Pair Group Method with Arithmetic Mean (UPGMA [99]) and Neighbour Joining [100], while the most popular of the latter are based on three approaches: Maximum Parsimony [101], Maximum Likelihood [102] and Bayesian Inference.

While character-based methods are sometimes considerably more accurate, distance-based methods are much faster to compute, and there is at least some evidence to suggest that Neighbour Joining is sufficiently accurate for closely related species [103]. Therefore, Neighbour Joining is used to build the trees in the SeqRank workflow, more specifically the EMBASSY[5] versions 'fdnadist' and 'fneighbor' of the classic PHYLIP tools 'dnadist' and 'neighbor'[6]. Clustal Omega [104] is used for fast, scalable alignment of multiple molecular sequences. After tree building, the tree is returned to the SeqRank workflow in Newick tree format, and drawn on a canvas on the web page itself. Tree drawing is performed by the open source Javascript library Jstree[7].

6.4 Technical Architecture

The SeqRank workflow is completely integrated into the StrainInfo platform. Its high-level architecture, depicted in Figure 6.8 therefore will look familiar to readers of Chapter 2. All communication with the client is done through the StrainInfo website. One minor difference, which will be discussed below, is that the SeqRank workflow is confined to a single page of StrainInfo. While complex communication occurs, the client never navigates away during execution of the workflow, except when the client forces such navigation. This greatly simplifies state management, as it can be done in client code, all within the context of the original page. Dependencies, shown as arrows in Figure 6.8, indicate that the poset ranking code involved in the SeqRank workflow is organised similarly to other StrainInfo modules. The SeqRank workflow web application

[5] `http://emboss.sourceforge.net/`
[6] `http://evolution.genetics.washington.edu/phylip.html`
[7] `http://lh3lh3.users.sourceforge.net/jstree.shtml`

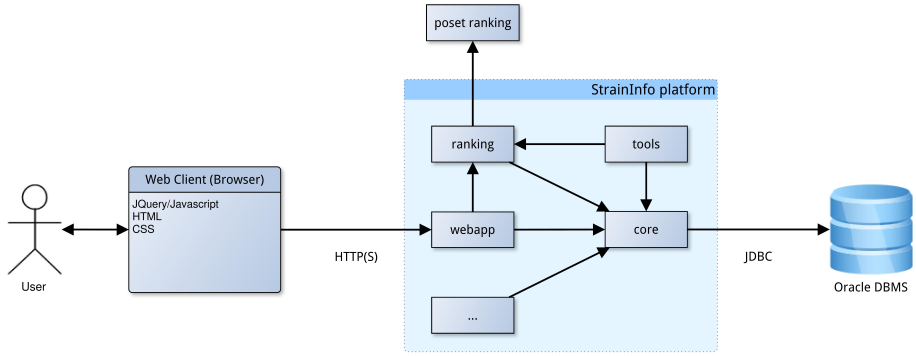


Figure 6.8: Overview of the SeqRank workflow architecture. Rectangle nodes indicate software modules and libraries. Arrows indicate direction of software dependencies or communication channels.

functionality, such as input parsing and state management, is integrated into the 'webapp' module, while poset ranking related functionality is abstracted into the 'ranking' module. Code specific to poset ranking, which can be used separately from the StrainInfo platform, is packaged and used as part of a separate poset ranking library. Shared functionality and the database access layer, as well as caching functionality, are completely abstracted through the 'core' module, allowing flexible reimplementation of data access, as well as re-use of functionality among modules.

6.4.1 Front-end layer

As discussed in previous sections, the SeqRank workflow is completely integrated into the StrainInfo web site and platform. It therefore closely follows look and feel of StrainInfo, and relies on the same standard web technologies, namely HyperText Markup Language (HTML) generated server-side from Java Server Pages (JSP) templates, with Cascading Style Sheets (CSS) providing rules for page layout and display, and dynamic functionality implemented in JavaScript supported by the jQuery library[8], which provides intuitive page manipulation functionality and a cross-browser compatibility layer, as well as animation functionality. Application code further relies on several extra

[8] `http://jquery.com/`

libraries, namely the previously mentioned Jstree for phylogenetic tree drawing, 'explorer canvas'[9] to support this tree drawing in older versions of Internet Explorer, and jQuery UI[10], a set of plugins built on top of jQuery that provide more advanced animations.

It was decided early on in development that to be truly effective as a decision support tool, the SeqRank workflow should allow easy access to wide range of information. Some of this exported information, such as sequence exports or the results of poset ranking, can be quite large or have long computation times. It is therefore infeasible to load all of this information on the first page load. For this reason, the workflow was designed as a single page application, embedded into StrainInfo: from first start of a run of the application, JavaScript code is activated that manages the view presented to the user (the current state of the page displayed in a browser), and runs further queries to the server. Most of this is done asynchronously. In other words, the page remains responsive, allowing the user to examine current results, while network requests to the application server continue in the background. Figure 6.9 illustrates how this works in practice. The initial page load by the client browser loads the page, including all related resources such as CSS layout code and JavaScript application code, in the form of the module file `selector.js`. It is this code that manages application state and requests further data from the server. The responses to these requests, when ready, are sent as HTML fragments, which are loaded into the page by request of the user (by selecting the relevant workflow stage). A typical example of such a fragment is shown in Listing 6.1. Therefore, templates and all relevant processing is done server-side, while front-end code manages client application state and can focus on data display and functionality. This, in combination with the reliance on already included libraries such as jQuery, keeps the JavaScript code size to an absolute minimum: `selector.js` only contains about 450 lines of code, which handle all client-side aspects of the SeqRank workflow operation.

6.4.2 Request/Response architecture

Some long-running workflow actions detailed in Section 6.3 may take quite long to complete. For instance, poset ranking of a full genus requires queries and computations that, when not cached and under load, may take up to 10

[9] `https://code.google.com/p/explorercanvas/`
[10] `http://jqueryui.com`

```
<div id="outlier">
  <table id="speciesseqtable" class="resultstable">
    <thead>
      <tr>
        <th>Species</th>
        <th>Strains</th>
        <th>16S</th>
        <th>Location</th>
      </tr>
    </thead>
    <tr>
      <td>
        <a href="/taxa/370125">
          <span class="speciesname">
            <em>Acidicaldus organivorans</em>
          </span>
        </a>
      </td>
      <td>
        <div class="strainnumber">
        <span><a href="/strains/689407">Y008</a></span>[...]
        </div>
      </td>
      <td>
        <a href="/sequences/AY140238">
          <span class="accession">AY140238</span>
        </a>
      </td>
      <td class="location">&lt;1..&gt;1340</td>
    </tr>
  </table>
</div>
```

Listing 6.1: Example of a typical HTML fragment returned by the SeqRank workflow web application, in this case the result of outlier selection, which is returned as a single table. Some of the non-essential HTML markup of this fragment was omitted for brevity. When requested, this fragment is loaded directly into the results section of the display page.

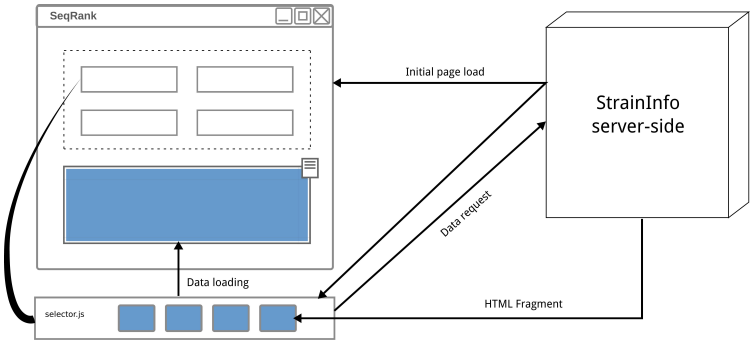


Figure 6.9: Front-end operation. The initial page request loads both a main page template, as well as JavaScript code (`selector.js`) to manage page state. Subsequent requests are generated by `selector.js` and return page fragments, which are stored client-side and slotted into place in the results view upon user request or at pre-defined times during application runtime.

minutes. Likewise, multiple alignment of very large selections of sequences can easily take tens of minutes. In the meantime, the application must remain responsive and not waste server resources. It is therefore not possible for some tasks to simply launch a request and wait for the result, as the long waiting involved will consume server processing threads as well as a network connection socket, and the connection itself may even time out, requiring complicated error handling to detect. Instead of relying on such handling, and tying up network resources in the process, application code can use infrequent *polling* to determine the current status of any workflow task. Polling is done over a network connection to the web server through regular HyperText Transfer Protocol (HTTP) connections. Appropriate HTTP status codes allow the server to signal task state to the client, and indicate when the body of a response contains the final result of processing.

The sequence diagram in Figure 6.10 illustrates how the polling system works in practice. A data handler is triggered during execution of the application to upload data to the server and requests results for a particular task, in this case the multiple alignment of selected sequences. Such an alignment will often require at least a minute of server-side processing time. Therefore, the poller subsystem initiates a polling pattern, using a generic utility function which launches an asynchronous request. A lightweight handler function processes the response to that request, scheduling a new request at a later

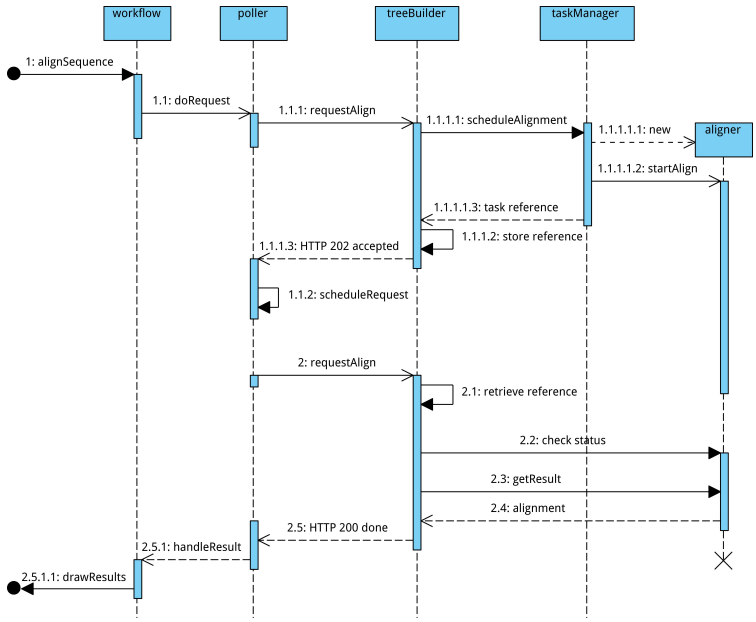


Figure 6.10: Sequence diagram of the typical request/response pattern used to poll for the status of long-running tasks in the SeqRank workflow. This example details how multiple sequence alignments are requested from the server. All HTTP requests are executed asynchronously (indicated by an open arrow).

time if necessary. The polling subsystem handles all connection-level details, and provides the results provided by the server back to the original workflow task handler upon completion of its request, where they are used to redraw details on the web page. Consistency between requests is handled server side by storing a reference to requested task in the user session (in the figure, this is performed by the `'treeBuilder'` object, which handles phylogenetic tree building tasks). A HTTP status code with value 202 is returned, indicating that the task is still processing. The stored reference is used to examine results of the executing tasks when new requests come in, returning the end result to the application with HTTP status code 200 if successful.

6.4.3 Middleware

Recall from Figure 6.8 that server-side processing is spread over three different StrainInfo modules. The first, the `webapp` module, is the main StrainInfo web application module. It contains all web-related functionality, which consists mostly of `Action` classes (henceforth referred to as 'actions') that process incoming requests, and the templates used to generate HTML or text fragments for client-side processing. The `webapp` module also packages all client-side application code that is downloaded on first page access, as described above. In contrast, the `ranking` module does not depend in any way on being deployed in a web application context. It contains all functionality related to the SeqRank workflow, including a small set of tasks which perform the actual ranking, and a set of tasks involved in calling external programs to perform tree building. Finally, the `core` module is used by the two other modules to abstract access to external data sources. In this case, it mainly provides access to the underlying Oracle Database Management System (DBMS).

As mentioned in Chapter 2, the StrainInfo web application relies on the use of the Apache Struts 2 framework[11]. In Struts 2 configuration, it is possible to bundle a set of actions as a separate package. The SeqRank workflow makes use of this functionality to bundle a set of common results used to implement polling (discussed previously in Section 6.4.2), as well as defining a FASTA format result type not used in other parts of the StrainInfo web application. A set of common restrictions are also shared between actions (for example, restrictions in the maximum number of sequences), and error pages specific to

[11] `http://struts.apache.org/development/2.x/`

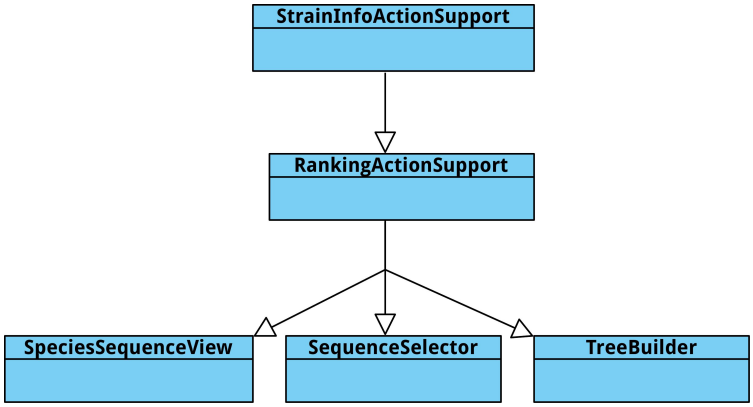


Figure 6.11: Class diagram of the most important SeqRank actions.

these results are configured in this package.

The most important classes involved in communication with the client are shown in Figure 6.11. Actions all have a common superclass, `RankingActionSupport`, itself derived from the `StrainInfoActionSupport` class that all StrainInfo action classes are derived from. The functionality is divided as such:

SpeciesSequenceView Species lookup (including rename history), as well as the second step in the workflow, which consists of linking said species to their type strains and 16S rRNA gene sequences.

SequenceSelector Gathers selected sequences and uses poset ranking services provided by the `ranking` module to rank sequences according to quality. The result is returned in the form of a table with the first ranked sequence pre-selected, and is used by the client application as input to the next step of the workflow.

TreeBuilder Manages tree building status and launches the tasks involved in tree building (multiple alignment, computing distance, building the tree itself) and returning the result to the client.

Fetching species, strain, and sequence information relies on access to the back-end database, which is performed by an independent database layer implemented in the `core` module. External resource access such as this is modelled using the Repository pattern [105], which abstracts away low-level

database details. This functionality is extremely similar to that involved in constructing passport pages in other areas of StrainInfo. Indeed, it reuses a lot of the same functionality, and will not be discussed here any further.

The `SequenceSelector` and `TreeBuilder` classes additionally rely on a number of long-running services specific to the SeqRank workflow. These are the ranking and tree building tasks defined in the `ranking` module, both of which rely on the ability to start such a task asynchronously and the ability to infrequently check the status of this process, in order to return results to an outside request when the process completes. In both cases, a separate application thread is necessary to run the computation process, regardless of whether it is running internally in the web application memory space, externally as a separate application on the same server, or even as a remote web service. This is necessary since starting and running this application process may otherwise take up all web application threads, completely blocking incoming requests (even those for unrelated areas of the StrainInfo website). Therefore, hand-off of the task must happen as quickly as possible, after which web the application threads can continue to process new incoming requests. There are some common requirements and pitfalls that must be satisfied for the implementation of this functionality, namely:

- Web applications typically exhibit 'bursty' behaviour, i.e. requests tend to cluster together, and so the web application encounters many long periods of low activity, interspersed with the occasional peak in demand. These peaks must not lead to unlimited allocation of new threads for asynchronous tasks, since this might well exhaust available memory resources on the server.

- It must be quick and easy to check whether computation has completed, as repeated polling makes this the most common task executed by workflow actions.

- Implementation should not involve low-level thread primitives, since their application can be extremely error-prone. Allowing web application code to manually start and manage threads may put the entire application at risk. It would also mean duplicating a lot of common functionality across several classes, which makes it much harder to maintain.

The first requirement can be addressed through the use of a *thread pool*: a

pool of worker threads that tasks get handed off to, and that can grow dynamically to a certain predefined maximum size. Worker threads are given work packets, which are implemented according to the 'Command' pattern [106]. In its simplest version, the worker thread simply calls a single function on the work packet object it is passed, which leads to the necessary actions. In Java, the `Runnable` interface defines similar behaviour. An instance of this interface can be passed to a manually created thread, or separate worker implementation could execute any `Runnable` implementation passed to it. The latter two requirements in the above list do require, however, that there is some way to ascertain the current status of the executing task, and a safe way to obtain the results. Transferring results from one thread to another must be done in a safe way, the rules of which can be complicated and error-prone to follow [107]. Fortunately, the Java Concurrency Framework, introduced in Java 5.0, greatly reduces effort involved in building such task-central parallel implementations. By introducing an `ExecutorService`, it provides a simple abstraction for thread pools, but also allows for status checking and result collection with some high level building blocks in the form of the `Callable` (which can be used instead of `Runnable`) and `Future` interfaces.

Listing 6.2, adapted from [107], shows some of the main features of these two interfaces, as well as the `TaskManager` interface, which is a simple wrapper around a configured `ExecutorService` for use within StrainInfo, bundled as part of the `core` module. `Callable` is quite similar to `Runnable`, except for the fact that an implementing task can return a result, or throw an exception. These extra features, however, are what allow an `ExecutorService`, or in this case a `TaskManager` to return an object of type `Future<V>` when a task is submitted (`V` being the parametric type of the task's result). Note from the code in Listing 6.2 that a `Future` can be interrogated about the current status of a task (e.g. through the `isDone()` method), and that it can be eased to (synchronously) retrieve the results, at any point during its lifetime. When a task is interrupted by an exception, this exception will be propagated to the caller of the `get()` function, ensuring that errors can be handled.

The presence of such a highly abstracted task execution framework makes it fairly straightforward to implement the sort of long-running workflow tasks necessary within the SeqRank workflow. Both for sequence ranking and for each step in tree building, this can be accomplished by creating an implementation of the `Callable` interface. In the current implementation, front-end actions query the database repository beforehand for all necessary information

```
public interface Callable<V> {
    V call() throws Exception;
}

public interface Future<V> {
    boolean cancel(boolean mayInterruptIfRunning);
    boolean isCancelled();
    boolean isDone();
    V get() throws InterruptedException, ExecutionException,
                   CancellationException;
    [...]
}

public interface TaskManager {
    public Future<V> submit(Callable<V> callable);
}
```

Listing 6.2: `Callable`, `Future`, and `TaskManager` interface

for the task to run, and construct a single instance responsible for the calculations of a particular workflow step for one particular client. Removing the requirement for database or network access further simplifies tasks sufficiently, so that their implementation is well focused on the task at hand. Once constructed, they are passed off to a `TaskManager` using the `submit()` function. The resulting `Future` result object can be used to query the task for its current status, provided a reference to it is available upon subsequent requests from the client. Therefore, the future itself is stored within the client session under a generated key that can be retrieved when receiving new requests. Web application actions then use the `isDone()` function of the retrieved `Future` object. When this method signals completion, `get()` is called to retrieve the results. Any exceptions generated during the running time of the instance of `Callable` are returned at this point and converted to an appropriate network response to be interpreted by the client (e.g. a HTTP status code in the range 500–599, indicating server error, or in the 400–499 range, possible when a malicious client submits faulty input data).

6.4.4 Caching Architecture

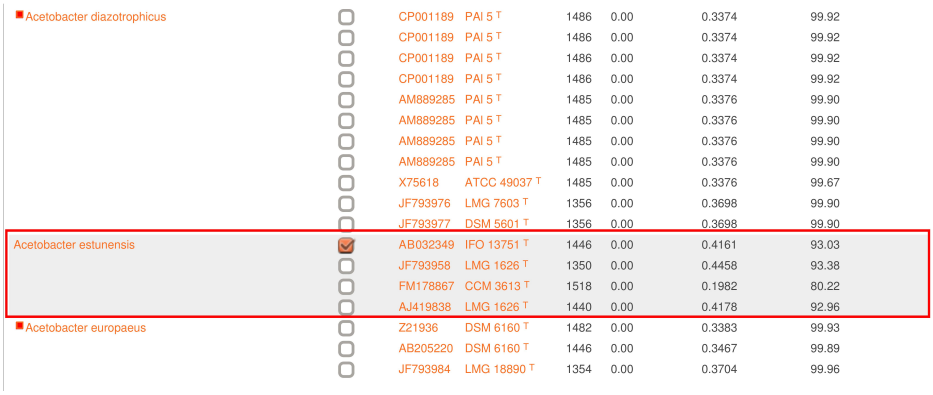
At the time of writing, StrainInfo is able to link slightly less than 11,000 species to their respective type strains. For each type strain, a poset may be pre-calculated for use by all clients, as long as the list of available 16S rRNA gene sequences for that type strain remain the same. Such a relatively limited number of rankings can easily be stored in a data repository or even in memory within the application if memory is abundant. As it makes little sense to recompute rankings, when only a small fraction of these posets actually need to be recomputed (through changes in type strain information or updates in their 16S rRNA gene sequences), SeqRank includes a ranking cache. The cache is updated automatically when requests are processed, and an offline tool periodically clears stale entries and updates the cache. At the time of writing, this tool was running weekly.

The ranking cache needs to include enough detail to check validity of an entry. Therefore, an entry in the cache contains a list of experiment identifiers (which correspond to accession numbers) and feature identifiers, with the (type) strain they were linked to, and the results of poset ranking in the form of a rank assigned to each sequence. These results include all metadata used in ranking. It is therefore possible to completely reconstruct the poset used to perform the ranking, and to show the values of sequence-derived attributes such as ambiguity percentage without having to recalculate them. An entry in the cache is invalidated if a sequence record has been removed or added, or if any of the sequence records have been updated since the caching of this entry. The latter can be checked easily using the last modification date included on all EMBL sequence records imported into the StrainInfo database.

6.5 Discussion

6.5.1 Case study: examining the genus *Acetobacter*

One of the most interesting aspects of the SeqRank workflow is that it exposes a lot of information on taxonomy, strains and 16S rRNA gene sequences that may be known to experts only, but is not easily accessible for the casual user. This section explores one example of how the parameters shown in the SeqRank workflow results, as well as the tree visualisation, make it much easier to examine a few typical data consistency problems. The example chosen for



Acetobacter diazotrophicus	☐	CP001189	PAI 5 T	1486	0.00	0.3374	99.92
	☐	CP001189	PAI 5 T	1486	0.00	0.3374	99.92
	☐	CP001189	PAI 5 T	1486	0.00	0.3374	99.92
	☐	CP001189	PAI 5 T	1486	0.00	0.3374	99.92
	☐	AM889285	PAI 5 T	1485	0.00	0.3376	99.90
	☐	AM889285	PAI 5 T	1485	0.00	0.3376	99.90
	☐	AM889285	PAI 5 T	1485	0.00	0.3376	99.90
	☐	AM889285	PAI 5 T	1485	0.00	0.3376	99.90
	☐	X75618	ATCC 49037 T	1485	0.00	0.3376	99.67
	☐	JF793976	LMG 7603 T	1356	0.00	0.3698	99.90
	☐	JF793977	DSM 5601 T	1356	0.00	0.3698	99.90
Acetobacter estunensis	☑	AB032349	IFO 13751 T	1446	0.00	0.4161	93.03
	☐	JF793958	LMG 1626 T	1350	0.00	0.4458	93.38
	☐	FM178867	CCM 3613 T	1518	0.00	0.1982	80.22
	☐	AJ419838	LMG 1626 T	1440	0.00	0.4178	92.96
Acetobacter europaeus	☐	Z21936	DSM 6160 T	1482	0.00	0.3383	99.93
	☐	AB205220	DSM 6160 T	1446	0.00	0.3467	99.89
	☐	JF793984	LMG 18890 T	1354	0.00	0.3704	99.96

Figure 6.12: Part of the results table for a SeqRank run on the type strains of the genus *Acetobacter*. There are 3 type strains shown in the picture, two of which have already moved and are not selected, but for which all derived 16S rRNA gene sequences show high average similarity. Not all cultures of the type strain of *Acetobacter estunensis* are this similar however, as the sequence with accession number FM178867 shows very low average similarity to the others.

this section is *Acetobacter*, a genus of the acetic acid bacteria (AAB), which is a family of bacteria that oxidize ethanol to acetic acid, or other alcohols to their respective acids or CO_2 [2].

Figure 6.12 shows that for certain type strains, such as that of *Acetobacter estunensis*, not all sequences are as highly similar as could be expected for 16S rRNA gene sequences of the same strain. In fact, the sequence derived from the culture with strain number CCM 3613^T shows only 80.22% average similarity to other sequences of the same strain. This might have several causes:

- Intra-genomic variability of 16S rRNA gene sequences. This does not appear to be a known event within the genus *Acetobacter*.

- Faulty strain information: StrainInfo might consider CCM 3613 a subculture of the type strain of *Acetobacter estunensis*, while it really is unrelated. Usually this is due to an annotation error. In this case, history information of the strain was manually curated by users, and clearly indicated on several BRC strain information pages, which makes this unlikely, though not impossible (Figure 6.13).

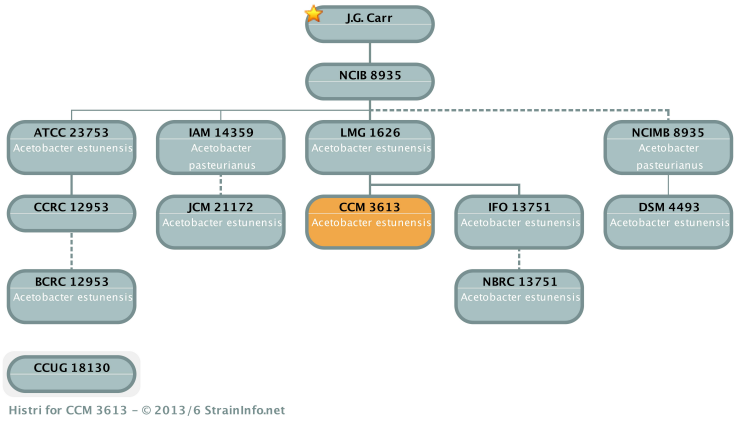


Figure 6.13: Histri of the *Acetobacter estunensis* type strain.

- A wrongly annotated 16S rRNA gene sequence record. It is always possible that the 16S rRNA gene sequence was submitted with errors in the metadata. In this case, the sequence record has both the species name and strain number clearly mentioned in its description. It may thus be a case of mistaken identity, but in that case it is an error that must necessarily have occurred at initial deposit of the sequence.

- Contamination of the original type strain during transfer to the CCM collection, or during sequencing of this culture.

Contamination and simple cases of mistaken identity (i.e. which sequence is derived from which strain) do occur occasionally. One of the central assumptions of the ranking algorithm in SeqRank is that such outliers should be excluded from ranking. The reasons for such outliers can be researched in several ways. One is to determine the transfer history of this particular strain. StrainInfo supports this in the form of the Histri trees, which are manually curated and accessible from the strain passport of any strain in StrainInfo. The histri found on the strain passport of CCM 3613^{T} (Figure 6.13) is almost complete and shows that CCM 3613^{T} was originally derived from LMG 1626^{T}, as is the only other culture for which a 16S rRNA gene sequence was found,

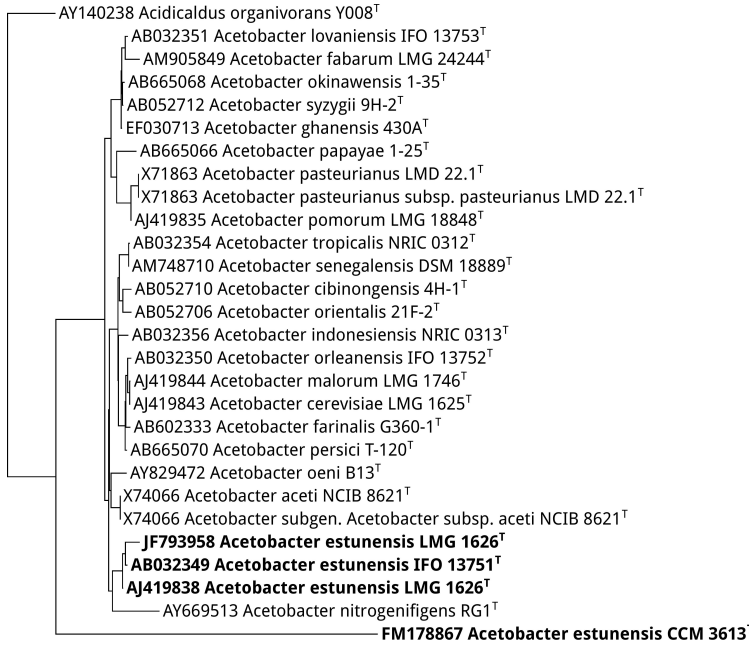


Figure 6.14: Phylogenetic tree from SeqRank for the genus *Acetobacter*. Labels of all 16S rRNA gene sequences of the *Acetobacter estunensis* type strain are highlighted with bold text. The sequence from culture *Acetobacter estunensis* CCM 3613^T (accession number FM178867) clearly diverges from other *Acetobacter* species, and specifically other subcultures of the same type strain.

IFO 13751^T. Since all three cultures are closely related, it is still not entirely clear where the problem might have occurred. It seems more likely that CCM 3613^T was contaminated during transfer, or that a problem occurred during its sequencing. But it is still possible that LMG 1626^T was contaminated, and that sequences from it and IFO 1375^T are actually the ones resulting from an unrelated strain.

One way microbiologists traditionally disentangle such a problem is by BLASTing the sequences to those found in INSDC. If one of the sequences shows high similarity to those of strains in a different genus, this might lead an expert to conclude that one is derived from the contaminant. Blasting the outlier against the non-redundant nucleotide collection of the NCBI shows its

high similarity to 16S rRNA genes from members of the genera *Bacillus* and *Lysinibacillus*. In the SeqRank workflow itself, the provided NJ tree can in some cases be used for the same purpose. The tree in Figure 6.14 has been generated by applying the SeqRank workflow to the genus *Acetobacter*, and manually including all 16S rRNA gene sequences found for the *Acetobacter estunensis* type strain. The tree shows that the three other sequences for *estunensis* cluster together in the tree and are much more closely related to other *Acetobacter* species than the original outlier, even when compared to the chosen outgroup. This further confirms that the sequence with accession number FM178867 must be the result of contamination, misannotation or other error. In this way, the SeqRank workflow provides an easy and intuitive way to examine and detect issues with sequence provenance within particular taxonomic groups. More advanced analysis can then of course be done using specialised tools, starting from the sequence exports provided by the workflow.

6.5.2 Applications

Casual identification based on 16S rRNA gene sequences is often attempted using simple alignment based search tools, based on the Basic Local Alignment Search Tool (BLAST) algorithm. While this often delivers a good first approximation of genus information, it is highly reliant on the quality of the search database, as well as the ability of the end user to correctly interpret the results. When species identification in the database are often wrong, and low quality sequences are present, use of BLAST is not sufficiently reliable for species identification [108]. It is also well known that a significant fraction of sequences in the central INSDC databases is misidentified or contains anomalies [84]. Species identification could also be done through the calculation of distance measures to existing, known strains, for which high-quality 16S rRNA gene sequence databases are a prerequisite.

For both approaches, the quality of the 16S rRNA gene sequence database can thus be considered crucial. Maintaining a database of high-quality 16S rRNA gene sequences for the genera under consideration is thus the first priority of phylogenetic researchers, or those that design and maintain the identification databases used in diagnostic identification. Since the SeqRank workflow can determine the current state of the target genus taxonomy, type strain information for each species in that genus, and at least one high-quality 16S rRNA gene sequence for each type strain, all from external and up-to-date resources,

it can be used to build such a database easily. Since it is based on the use of the type strain, it is also less sensitive than a simple BLAST query to certain types of annotation problems (such as misspelled or outdated species names). One possible downside of using the SeqRank workflow for identification database construction is that no manual curation was done on sequence information, which might result in incomplete strain information leading the user to use the wrong sequence. While this is also a danger with manually curated databases, which do occasionally contain errors, it is important that users double-check the results of any automated pipeline, and crossreference these with the results of databases such as LTP and GreenGenes.

Much of the same caveats apply to using the SeqRank workflow to assemble a list of sequences for use in taxonomic studies. Type strain and taxonomic information, including rename history, is collected from up-to-date resources, and as correct as it can be made within the context of StrainInfo. Data inconsistencies are possible, however. For instance, as mentioned in Chapter 2, a blacklist is maintained within StrainInfo of faulty externally discovered strain data. Such mistaken records may cause cultures from different strains to be considered part of the type strain, an error that could introduce 16S rRNA gene sequences from other strains into SeqRank output. It is here that SeqRank, too, requires some degree of manual monitoring.

An issue more fundamental to the science of taxonomy is that all taxonomic knowledge in SeqRank is based only on the current validly published names. These names can be considered to have some validity, but whether this taxonomy is correct is a matter that can only be answered by a taxonomic expert, be it on the specific taxon or in bacteriology as a whole. This makes it extremely critical that an expert building his or her own taxonomic database can intervene at set places in the algorithm, and especially before export of the resulting sequences. The inclusion of rename history in SeqRank is certainly helpful in this respect, as it quickly points out conflicting names in a very straightforward manner.

6.5.3 Future work

No software tool is ever truly finished. While the SeqRank workflow started as a proof of concept, it is now truly useful as a way to examine type strain 16S rRNA gene sequences in bulk, using a strain selection based on nomenclature and taxonomy. A seed alignment could be used to greatly increase

speed of the multiple alignment. New web technologies can be employed to improve the user interface, while more of the calculation work could be offloaded to separate servers when performing ranking or building trees. The latter could be accomplished using web service technologies already deployed within StrainInfo. Additionally, the SeqRank workflow shows the way for a number of improvements that may greatly increase its usability and its coverage of current sequence databases, as 16S rRNA gene sequences of type strains are just a start.

Extracting the highest quality sequences of a certain gene for a strain, however, is not a use case limited to type strains. It is straightforward to imagine studies that want to move beyond just type strains, for example, for comparative studies of the 16S rRNA gene of all known strains of a certain species. The same could then be done for different genes. While the number of type strains is manageable, there are currently close to 300,000 strains known to StrainInfo. Manually curating sequence selections for this amount of strains is simply not feasible, especially when expanded to other genes. The SeqRank workflow can in principle be expanded for such uses, though there are two main hurdles. The first is the specificity of the current ranking approach to other genes. Each new gene that is added requires construction of gene-specific ranking criteria and devising a way to test them. For coding genes, this would also mean that sequences must first be translated, and frame shifts must be detected, lest measures such as average similarity don't work at all. For this group of sequences it is equally likely that a further filtering step could be added to detect obvious defects in these genes. However, even with these extra requirements, much of the current pipeline can be reused, as well as some of the current criteria (such as the length criterion), and SeqRank itself would need only minimal alterations to provide the end user with a similar view of information for other genes.

The second hurdle in expanding this approach is the lack of current taxonomic information on many of the currently known strains. While StrainInfo has gathered taxonomic data from BRCs, many of the taxonomic labels attached to various strains are now outdated, or possibly faulty. It is here that most curation is required. The presence of a rename history within the StrainInfo database, implemented for use in the SeqRank application, can also be used to infer current taxonomic status of strains. Reconciling the species data of the subcultures of a strain in StrainInfo might pose its own problems, however, as it is more than possible that species information conflicts between

subcultures. On the other hand, this might also yield valuable data on the distribution of isolation studies, showing researchers what taxonomic groups are underrepresented in StrainInfo. It could also provide a great technique for consistency checking of strain data in StrainInfo. It is, after all, likely that if the species annotations on cultures from different BRCs greatly differ (i.e. they are annotated as belonging to extremely different taxonomic groups), that the strain information needs to be double checked by hand. Since mistakes do pop up infrequently in culture records, especially when it comes to references to other strain numbers, any mechanism that allows StrainInfo to target strains that need to be checked for faulty references can be extremely valuable, as it is infeasible to regularly and completely survey the entire extent of strain information carried by StrainInfo. The use of SeqRank measures such as the average similarity measure might well be employed as an additional reference in such an approach. Scoring strains as being 'suspicious' in this way could be a great approach for StrainInfo quality management, and something to consider integrating in the platform in the future, where it can be used by our own curators or users to discover data issues.

This chapter is an extended version of the publication:
De Smet, W., De Baets, B., De Loof, K., De Vos, and P., Dawyndt, P. (2013). "SeqRank: quality assessment and ranking of 16S rRNA gene sequences in StrainInfo". Application note submitted for publication.

7
Conclusions and outlook

The value of explicit links

Microbiological research is becoming an increasingly data-driven science, relying more and more on the availability of various kinds of data and metadata stored in disparate databases. It was shown in the first chapters that the StrainInfo platform was built on the linkage of strain numbers, assigned to strains by researchers and culture collections in their own database (culture catalogues). These strains can then be linked to other data sources, such as molecular sequence, taxonomic, and publication databases. The Genomic Rosetta Stone project provides another avenue by which different databases can be linked together, focusing on the linkage between sequenced genomes and databases that gather additional data. The StrainInfo platform has been built on top of these links as a way in which users can easily find and navigate data related to cultured microbial strains. It is a popular resource in its own right, as shown in Chapter 2, providing support for the practice of research in microbiology. The creation of the Genomic Rosetta Stone (GRS) shows one way in which future linked databases might be constructed, on the basis of explicit mapping of identifiers by data providers themselves.

Establishing links within StrainInfo reveals data inconsistency issues with

the databases being integrated. Aggregating cultures into strains has revealed the presence of errors in many culture catalogs. The same is true for listed strains in taxonomic databases and strain number annotations in the molecular sequence databases, where strain identity is sometimes mistaken. Many of these problems remain invisible because they traverse database boundaries, until explicit links are established between data sets. The development of a mashup comparison app in the context of the GRS is an illustration of this fact, showing that the approach of linking various databases on the basis of common identifiers can also be used to explicitly expose inconsistencies in the data content of those databases. As is clear from the chapters on data integration and the GRS, the main barriers to the development of applications on top of such cross-linked databases, is the availability of data and the format in which this data is available, often requiring heavy data processing before integration can be performed in valid or useful ways.

The development of the SeqRank workflow has enabled a closer look at the results of integration of taxon, sequence and strain data, by making use of it in a single application. This can reveal errors in existing data sets, as explored in Chapter 5, and can be further used to improve existing data sets. The SeqRank algorithm ensures that, in general, sets of high-quality representative 16S rRNA gene sequences can be created automatically within StrainInfo. By providing the parameters used by the algorithm to the users, we have for the first time created a completely verifiable approach to this problem, which previously was only performed manually according to subjective criteria. This approach can be used to find errors in databases, but also to keep generated databases up-to-date and to aid researcher in day-to-day tasks, such as to build a simple phylogenetic tree in order to examine the diversity of particular taxa. Furthermore, working at the strain level, SeqRank and the SeqRank workflow are likely to scale well in the future and remain useful, even where manual curation might in future not be tenable.

Outlook

There is a myriad of possible improvements to StrainInfo, GRS, and SeqRank. As is often the case in software development, the question is not whether any single addition would be useful, but which of them should be prioritized. StrainInfo can always benefit from having access to more data from more culture collections and better administration processes for the developers.

Improvements to the user experience are also in constant development and will continue to be so for the entire future of StrainInfo. Some changes in SeqRank, such as more feedback on the shape of the poset derived and the addition of a universal alignment of 16S rRNA gene sequences from one of the existing expert databases, seem straightforward and fit well within the scalable nature of SeqRank.

Original versions of the SeqRank workflow were not aware of the presence of basonyms, which may seem a minor point but proved to greatly reduce the usability of the tool to microbiologists in initial tests, as the structure of the current taxonomy tends to change quite frequently. The addition of better taxonomic information from LPSN into StrainInfo (the parsing process of which was mentioned in Chapter 3) was therefore a prerequisite for the launch of the SeqRank workflow into StrainInfo. Even so, StrainInfo's grasp on the history and current status of a taxonomic name is rudimentary at this point. There is no concept on the type of changes that may happen, so at this point it is hard to tell whether a species has been renamed, whether the spelling was adjusted, whether the type strain died and was replaced, whether it is the hetero- or homotypic synonym of another name and so on. All this data could be added to the StrainInfo database, though it may require StrainInfo developers to lead the way in finally digitizing much of the official taxonomic history. Re-exporting this data to the community in an easily readable format could further improve the handling of taxonomic information in other databases, creating a positive feedback loop where the integration of StrainInfo data in their data sets would reveal the errors and inconsistencies that would inevitably crop up.

The complete lack of consistency between database interfaces can make it unnecessarily complicated to perform the data integration on which new insights and quality checking routines could be built, even when new data is made freely available. The case study of the Genomic Rosetta Stone project showed that it is certainly possible to develop new applications on top of existing databases, as StrainInfo itself also demonstrates, but at greater cost than is strictly necessary. As evidenced by the MIxS standards [45], there is some support for better quality of the content of data records. The form in which data records can be requested, stored and processed is still a matter of debate, although the current trend seems to point to the use of some form of controlled vocabulary such as that provided by existing, often XML-based standards, such as the Genomic Contextual Data Markup Language (GCDML) [109] and the Microbiological Common Language (MCL) [14].

Finally, the SeqRank workflow, which has been the beneficiary of all this data integration and quality checking work (and lends itself to the same), is only just in its infancy. Since the 16S rRNA gene has proven to be an insufficiently precise phylogenetic marker for a range of bacterial and archaeal genera, the next big step for its development is the extension of the same approach to other housekeeping genes. This expansion can build on the functionality already defined for determining sets of sequences linked to (type) strains of certain taxonomic names, but it will require the establishment of new parameters by which the sequences may be ranked. As the housekeeping genes targeted will invariably include protein coding genes, these parameters may end up reliant on some sort of universal alignment in order to detect frame shifts and similar common problems, which is an extra problem that has to be solved before this approach can be made worthwhile. Nonetheless, the initial design of the same parameters for the 16S rRNA gene has shown ways in which the data itself can guide the parameter design. At this moment, the importance of the ranking algorithm itself may seem rather small, as not many sequences of the other housekeeping genes exist for most strains, the lowered cost and growing attention to full coverage sequencing of the entire genomes of known strains will continue to add more copies over time. Manually sifting through this data for the strains of a fast growing list of prokaryotic taxa, for a dozen or even only a few genes, will simply not be feasible. It is here that the framework laid by the original SeqRank workflow could greatly be expanded for future work, and that in my opinion the way forward lies. In future, other approaches to ranking may rise to the fore, but the general approach of strain-based data collection and some form of ranking method to select representatives will likely be relevant for a long time to come, as there will generally be a need to be able to compare genetic information from new studies to that of the known diversity in any study seeking to expand microbiological knowledge.

8

Nederlandstalige samenvatting

De uitwisseling van onderzoeksmateriaal is een belangrijk onderdeel van de wetenschappelijke traditie in de biologische wetenschappen. Enkel door uitwisseling van het origineel materiaal kunnen onderzoekers immers studies herhalen en op het vorige werk voortbouwen. Micro-organismen zoals bacteriën en archaea worden zo al sinds hun ontdekking uitgewisseld, en de latere ontdekking van methodes om deze organismen te kweken en te bewaren in het labo in zogenaamd pure culturen heeft deze traditie slechts versterkt. Ook van fungi, een grote groep van micro-organismen waaronder gisten, werden door onderzoekers over de jaren collecties opgebouwd waarvan de inhoud uitgewisseld werd.

8.1 StrainInfo

Veel van de bestaande gekende diversiteit bevindt zich in private onderzoekscollecties. Het opgroeien en verdelen van het biologisch materiaal vraagt echter een aanzienlijke investering van tijd en geld. Al snel zijn daarom ook publieke collecties van microbiële collecties ontstaan, welke nu voor een groot deel instaan voor het bijhouden en beschikbaar maken van de gekende microbiële diversiteit. Zulke zogenaamde Biological Resource Centers (BRCs) laten vaak

andere entiteiten (waaronder onderzoekers en bedrijven) toe nieuwe culturen van microbiële stammen in te brengen in hun collectie. Bij zo'n deposito wordt aan deze subcultuur een stamnummer toegekend. Onderzoekers kunnen ten allen tijde subculturen van de publieke beschikbare microbiële stammen bekomen van een BRC (tegen een vergoeding), waarbij het stamnummer gebruikt wordt om de gewenste stam te identificeren.

Aangezien ook BRCs stammen uitwisselen met elkaar zijn verschillende culturen van dezelfde stam vaak van verschillende collecties verkrijgbaar. Elk van deze culturen heeft dan een ander stamnummer, dat in een catalogus van de BRC opgelijst wordt. Doordat elke BRC zich typisch enkel bewust is van de geschiedenis van een stam op het punt dat deze toegevoegd werd aan de collectie, kan deze niet alle mogelijke stamnummers oplijsten waaronder de stam bechikbaar is. Hierdoor is het vaak moeilijk voor gebruikers om andere culturen van dezelfde stam te vinden, wat het veel duurder kan maken dan noodzakelijk om een stam te bestellen. Het gebrek aan een overzicht van alle stamnummers maakt het ook moeilijker om op zoek te gaan naar publicaties of gensequenties gebaseerd op deze stam, aangezien ze mogelijk verwijzen naar het bronmateriaal via een ander stamnummer. Door de stamnummers en de geschiedenis van hun uitwisseling te extraheren en expliciet te koppelen aan de informatie uit andere databanken, kan StrainInfo een zogenaamd 'strain passport' opbouwen, dat alle gekende stamnummers van de culturen van een stam verzamelt en toont, samen met alle gekende sequenties en publicaties die ervan afgeleid zijn. Die extra informatie is vaak interessant op zich en krijgt ook zijn eigen 'publication passport' (voor publicaties), 'taxon passport' (voor taxonomische namen), of 'sequence passport' (voor genetische sequenties). Op elk van deze pagina's worden dan weer de daaraan gekoppelde publicaties, stammen of sequenties getoond, wat de microbioloog een duidelijk, makkelijk navigeerbaar overzicht informatie gerelateerd aan een microbiële stam geeft.

8.2 Het Genomic Rosetta Stone project

De kosten voor het bepalen van genetische sequenties zijn de laatste jaren sterk gedaald, met als gevolg dat steeds meer studies gebruik maken van sequenties van het volledige genoom van organismen. Dit geldt in het bijzonder voor bacteriën en archaea, die een relatief klein genoom hebben. Vele data providers voorzien extra informatie over, en analyses van, deze genoomsequenties. Zo

bevat StrainInfo, bijvoorbeeld, informatie over het bronmateriaal (de stam waarvan het genoom afkomstig is), terwijl andere bronnen bestaan voor de evaluatie van de kwaliteit van sequenties of voor het oplijsten van de lokaties waarvan het bronmateriaal afkomstig was. Het Genomic Rosetta Stone (GRS) project koppelt al deze databanken met behulp van open standaarden en informatie vanuit de providers zelf. Het werk in deze thesis rond GRS biedt steun voor de creatie van nieuwe, innovatieve applicaties gebaseerd op de data van verschillende data providers. Het liet ook toe om deze databanken beschikbaar te maken vanuit StrainInfo via een 'genome passport' en een 'genome browser'.

8.3 De SeqRank workflow

In de fylogenie wordt de evolutionaire geschiedenis van (microbiële) organismen bestudeerd op basis van genetische informatie. Voor bacteriën en archaea is het 16S rRNA gen hiervoor veruit het meest gebruikte. Vaak zijn er per stam verschillende kopieën van de sequenties van dit gen beschikbaar in de publieke databanken. Niet al deze kopieën zijn altijd van even goede kwaliteit, waardoor het nodig wordt voor diversiteit- en taxonomiestudies om voor elke stam uit de verschillende opties een enkele representatieve sequentie van hoge kwaliteit te selecteren. Het bestaan van expliciete links tussen sequenties, culturen (stamnummers) en stammen laat toe om vanuit StrainInfo nieuwe applicaties te bouwen die het mogelijk maken deze sequenties automatisch terug te vinden. Door de integratie van een rangschikkingsalgoritme gebaseerd op poset ranking kunnen representatieve sequenties eveneens automatisch gekozen worden. Deze functionaliteit werd geïntegreerd in StrainInfo als SeqRank, dat alle 16S rRNA gensequenties beschikbaar voor een bepaalde stam of typestam rangschikt naar hun kwaliteit.

Onderzoekers dienen vaak representatieve sequenties te kiezen voor de typestammen van meerder taxonomische groepen. De SeqRank workflow laat dit toe door SeqRank toe te passen op de typestam van elke soort of deelsoort gevonden onder een taxonomische groep op het niveau van genus of hoger. De workflow laat toe om lijsten van representatieve sequenties makkelijk en snel te genereren of bij te werken. Door de expliciete connecties in StrainInfo tussen taxon, cultuur, stam en taxon, kan de SeqRank workflow ook gebruikt worden om fouten te vinden in bestaande collecties van 16S rRNA gensequenties. De

SeqRank workflow biedt als groot voordeel dat deze toegepast kan worden op volledig automatische manier, en dat de verschillende criteria gebruikt in het SeqRank algoritme aangepast kunnen worden voor hun toepassing op andere genen. Dit maakt de workflow inherent schaalbaar, zelfs als deze taak, door het almaar toenemende volume sequentiedata, onmogelijk nog handmatig kan uitgevoerd zou kunnen worden.

9 Summary

The exchange of research material is an important part of scientific tradition in the biological sciences. Only by exchanging the original material can researchers can repeat and build on previous work. Microorganisms such as bacteria and archaea have been exchanged in this way since their discovery, and the later discovery of methods to grow and store these organisms in a laboratory environment in so-called pure cultures have only strengthened that tradition. Research in Fungi, another large group of of microorganisms that includes, among many others, the various yeasts, has also greatly benefited from the existence of collections built by researchers over the years.

9.1 StrainInfo

Much of the existing known diversity is located in private research collections. Growing and distributing biological material, however, requires a significant investment of time and money. As a result, public service collections for microbial material were soon created. These collections are now largely responsible for preserving and making available the known microbial diversity. Such so-called Biological Resource Centers (BRCs) often allow other entities (including researchers and companies) to deposit new cultures of microbial strains into

their collection. Each deposit is assigned a standard number, called a strain number. Researchers can at any time obtain subcultures of publicly available microbial strains from obtain a BRC (for a fee), wherein the strain number serves as a to uniquely identify the desired strain

Since BRCs often exchange microbial material between themselves, cultures of the same strain are commonly available from multiple collections. Each of these cultures is assigned a different strain number that is listed in the catalog of the BRC. Because each BRC is typically only aware of the exchange leading up to accession into its collection, it cannot list all possible strain numbers under which the strain is available. This often makes it difficult for users to find different cultures of the same strain, which can make it much more expensive than necessary to order a strain. The lack of a list of all strain numbers also makes it harder to look up publications or gene sequences based on this strain, as they may refer to the source material by one of the other strain numbers. By extracting all known strain numbers and explicitly linking them to the information found in other databases, StrainInfo has created a so-called 'strain passport', which lists all known strain numbers of the cultures of a strain, along with all sequences and publications known to be derived from it. The extra information is often interesting in its own right and is thus also afforded a 'publication passport' (for publications), 'taxon passport' (for taxonomic names), or 'sequence passport' (for genetic sequences). On each of these pages the associated publications, strains or sequences are shown, providing the microbiologist with a clear, easily navigable overview of microbial strain related information.

9.2 The Genomic Rosetta Stone project

The costs of genetic sequencing have fallen greatly in the past few years. More and more studies are therefore making use of full genome sequences. This is true in particular for bacteria and archaea, that possess a relatively small genome. Many data providers provide additional information on, and analysis of, these genome sequences. StrainInfo, for example, contains information about the source material (the strain from which the genome was extracted), while other sources exist for the evaluation of the quality of sequences, or to lists the locations from which source material was isolated. The Genomic Rosetta Stone (GRS) project links these databases using open standards and

published information provided by from the providers themselves. The work on the GRS described in this thesis provides for support for the creation of new, innovative applications based on the data from various data providers. It has also allowed for the integration of these databases in StrainInfo via the creation of a 'genome passport' and a 'genome browser'.

9.3 The SeqRank workflow

In phylogeny, the evolutionary history of (microbial) organisms is studied on the basis of genetic information. For bacteria and archaea, the 16S rRNA gene is by far the most widely used gene for this purpose. For many strains, multiple copies of the sequences of this gene are available from the public databases. Not all of these copies are always of good quality, making it necessary for diversity and taxonomy studies to select a single representative and high-quality 16S rRNA gene sequence for every included strain. The existence of explicit links between sequences, cultures (strain numbers) and strains allows the construction of new applications within StrainInfo to retrieve such gene sequence records automatically. By integrating a ranking algorithm based on poset ranking, representative sequences can be chosen automatically. This functionality was integrated into StrainInfo as SeqRank, which ranks all 16S rRNA gene sequences of a particular strain or type strain according to look up the attributes List on it and rank them on quality.

Researchers often need to choose representative 16S rRNA gene sequences type strains of several taxonomic groups. The SeqRank workflow allows them to accomplish this task by applying SeqRank to the type strain of each species or sub-species found within a taxonomic group at the rank of genus or above. The workflow can be used to quickly generate or update lists of representative sequences. The explicit connections between taxon, culture, strain and taxon provided by StrainInfo also enable the use of the SeqRank workflow as a tool for finding errors in existing collections of 16S rRNA gene sequences. The SeqRank workflow offers the great advantage that it operates fully automatically, and that and that the different criteria used in the algorithm can be adjusted to different genes. These qualities make the workflow inherently scalable, even when the ever increasing volume of publicly available sequence data will eventually preclude the manual execution of such tasks.

Bibliography

[1] Whitman, W.B., Coleman, D.C., and Wiebe, W.J. (1998). "Prokaryotes: The unseen majority". *Proceedings of the National Academy of Sciences*, **95**: 6578–6583.

[2] Madigan, M.T., Martinko, J.M., Dunlap, P.V., and Clark, D.P., *Brock Biology of microorganisms 12th edn.* (Pearson Benjamin Cummings, 2009).

[3] Woese, C.R. and Fox, G.E. (1977). "Phylogenetic structure of the prokaryotic domain: The primary kingdoms". *Proceedings of the National Academy of Sciences*, **74**(11): 5088–5090. doi:10.1073/pnas.74.11.5088.

[4] Woese, C., Kandler, O., and Wheelis, M. (1990). "Towards a natural system of organisms: proposal for the domains archaea, bacteria, and eucarya". *Proceedings of the National Academy of Sciences*, **87**(12): 4576–4579. doi:10.1073/pnas.87.12.4576.

[5] Pace, N.R. (2009). "Mapping the tree of life: progress and prospects". *Microbiology and molecular biology reviews*, **73**(4): 565–576. doi:10.1128/MMBR.00033-09.

[6] Karsch-Mizrachi, I., Nakamura, Y., and Cochrane, G. (2012). "The International Nucleotide Sequence Database Collaboration". *Nucleic Acids Research*, **40**(Database issue): D33–D37. doi:10.1093/nar/gkr1006.

[7] Janssens, D., Arahal, D.R., Bizet, C., and Garay, E. (2010). "The role of public biological resource centers in providing a basic infrastructure for microbial research". *Research in Microbiology*, **161**(6): 422–429. doi:10.1016/j.resmic.2010.03.009.

[8] Tindall, B.J., Kämpfer, P., Euzéby, J.P., and Oren, A. (2006). "Valid publication of names of prokaryotes according to the rules of nomenclature: past

history and current practice". *International Journal of Systematic and Evolutionary Microbiology*, **56**(Pt 11): 2715–2720. doi:10.1099/ijs.0.64780-0.

[9] OECD, *Biological Resource Centres: Underpinning the future of life sciences and biotechnology* (OECD Publishing, 2001). ISBN 9789264186903. doi:10.1787/9789264193550-en.

[10] Dawyndt, P., Vancanneyt, M., De Meyer, H., and Swings, J. (2005). "Knowledge accumulation and resolution of data inconsistencies during the integration of microbial information sources". *IEEE Transactions on Knowledge and Data Engineering*, **17**: 1111–1126.

[11] Euzéby, J. (1997). "List of Bacterial Names with Standing in Nomenclature: a folder available on the Internet". *International Journal of Systematic Bacteriology*, **47**(2): 590–592.

[12] Verslyppe, B. (2012). *StrainInfo: from microbial information to microbiological knowledge*. Ph.D. thesis, Ghent University.

[13] Dijkshoorn, L., Ursing, B.M., and Ursing, J.B. (2000). "Strain, clone and species: comments on three basic concepts of bacteriology". *Journal of Medical Microbiology*, **49**(5): 397–401.

[14] Verslyppe, B., Kottmann, R., De Smet, W., De Baets, B., De Vos, P., and Dawyndt, P. (2010). "Microbiological Common Language (MCL): a standard for electronic information exchange in the microbial commons". *Research in Microbiology*, **161**(6): 439–445.

[15] Verslyppe, B., De Smet, W., De Baets, B., De Vos, P., and Dawyndt, P. (2011). "Make Histri: reconstructing the exchange history of bacterial and archaeal type strains". *Systematic and Applied Microbiology*, **34**(5): 328–36.

[16] Vandamme, P., Pot, B., Gillis, M., de Vos, P., Kersters, K., and Swings, J. (1996). "Polyphasic taxonomy, a consensus approach to bacterial systematics". *Microbiological Reviews*, **60**(2): 407–438.

[17] Lapage, S., Sneath, P., Lessel, E., Skerman, V., Seeliger, H., and Clark, W., editors, *International Code of Nomenclature of Bacteria: Bacteriological Code, 1990 Revision* (ASM Press, Washington (DC), 1992). ISBN 155581039X.

[18] McNeill, J., Barrie, F., Buck, W., Demoulin, V., Greuter, W., Hawksworth, D.L., Herendeen, P.S., Knapp, S., Marhold, K., Prado, J., Prud'homme van Reine, W.F., Smith, G., Wiersema, J.H., and Turland, N.J., editors, *International Code of Nomenclature for algae, fungi, and plants (Melbourne Code)* (Koeltz Scientific Books, 2012). ISBN 978-3-87429-425-6. Adopted by the Eighteenth International Botanical Congress Melbourne, Australia, July 2011.

[19] Klenk, H.P. and Göker, M. (2010). "En route to a genome-based classification of Archaea and Bacteria?" *Systematic and Applied Microbiology*, **33**(4): 175–182.

[20] Wu, D., Hugenholtz, P., Mavromatis, K., Pukall, R., Dalin, E., Ivanova, N.N., Kunin, V., Goodwin, L., Wu, M., Tindall, B.J., Hooper, S.D., Pati, A., Lykidis, A., Spring, S., Anderson, I.J., D'haeseleer, P., Zemla, A., Singer, M., Lapidus, A., Nolan, M., Copeland, A., Han, C., Chen, F., Cheng, J.F., Lucas, S., Kerfeld, C., Lang, E., Gronow, S., Chain, P., Bruce, D., Rubin, E.M., Kyrpides, N.C., Klenk, H.P., and Eisen, J.A. (2009). "A phylogeny-driven genomic encyclopaedia of Bacteria and Archaea". *Nature*, **462**(7276): 1056–60.

[21] Crous, P.W., Gams, W., Stalpers, J.A., Robert, V., and Stegehuis, G. (2004). "MycoBank: an online initiative to launch mycology into the 21st century". *Studies in Mycology*, **50**: 19–22.

[22] Wheeler, D.L., Barrett, T., Benson, D.A., Bryant, S.H., Canese, K., Chetvernin, V., Church, D.M., DiCuccio, M., Edgar, R., Federhen, S., Geer, L.Y., Kapustin, Y., Khovayko, O., Landsman, D., Lipman, D.J., Madden, T.L., Maglott, D.R., Ostell, J., Miller, V., Pruitt, K.D., Schuler, G.D., Sequeira, E., Sherry, S.T., Sirotkin, K., Souvorov, A., Starchenko, G., Tatusov, R.L., Tatusova, T.A., Wagner, L., and Yaschenko, E. (2007). "Database resources of the National Center for Biotechnology Information". *Nucleic Acids Research*, **35**(Database issue): D5–D12. doi:10.1093/nar/gkl1031.

[23] Woese, C.R. (1987). "Bacterial evolution". *Microbiological Reviews*, **51**(2): 221–71.

[24] Wayne, L., Brenner, D., Colwell, R., Grimont, P., Kandler, O., Krichevsky, M., Moore, L., Moore, W., Murray, R., Stackebrandt, E., Starr, M., and

Trüper, H. (1987). “Report of the Ad Hoc Committee on Reconciliation of Approaches to Bacterial Systematics”. *International Journal of Systematic Bacteriology*, **37**(4): 463–464.

[25] Stackebrandt, E. and Goebel, B.M. (1994). “Taxonomic note: a place for DNA-DNA reassociation and 16S rRNA sequence analysis in the present species definition in bacteriology”. *International Journal of Systematic Bacteriology*, **44**(4): 846–849.

[26] Sboner, A., Mu, X.J., Greenbaum, D., Auerbach, R.K., and Gerstein, M.B. (2011). “The real cost of sequencing: higher than you think!” *Genome Biology*, **12**(8): 125. doi:10.1186/gb-2011-12-8-125.

[27] Benson, D.A., Karsch-Mizrachi, I., Lipman, D.J., Ostell, J., and Sayers, E.W. (2010). “GenBank”. *Nucleic Acids Research*, **38**(Database issue): D46–D51. doi:10.1093/nar/gkp1024.

[28] Cochrane, G., Alako, B., Amid, C., Bower, L., Cerdeño Tárraga, A., Cleland, I., Gibson, R., Goodgame, N., Jang, M., Kay, S., Leinonen, R., Lin, X., Lopez, R., McWilliam, H., Oisel, A., Pakseresht, N., Pallreddy, S., Park, Y., Plaister, S., Radhakrishnan, R., Rivière, S., Rossello, M., Senf, A., Silvester, N., Smirnov, D., Ten Hoopen, P., Toribio, A., Vaughan, D., and Zalunin, V. (2013). “Facing growth in the European Nucleotide Archive”. *Nucleic Acids Research*, **41**(Database issue): D30–D35. doi:10.1093/nar/gks1175.

[29] Ogasawara, O., Mashima, J., Kodama, Y., Kaminuma, E., Nakamura, Y., Okubo, K., and Takagi, T. (2013). “DDBJ new system and service refactoring”. *Nucleic Acids Research*, **41**(Database issue): D25–D29. ISSN 1362-4962. doi:10.1093/nar/gks1152.

[30] Pruesse, E., Quast, C., Knittel, K., Fuchs, B., Ludwig, W., Peplies, J., and Glöckner, F. (2007). “SILVA: a comprehensive online resource for quality checked and aligned ribosomal RNA sequence data compatible with ARB”. *Nucleic Acids Research*, **35**(21): 7188–7196.

[31] Quast, C., Pruesse, E., Yilmaz, P., Gerken, J., Schweer, T., Yarza, P., Peplies, J., and Glöckner, F.O. (2013). “The SILVA ribosomal RNA gene database project: improved data processing and web-based tools”. *Nucleic Acids Research*, **41**(Database issue): D590–D596. doi:10.1093/nar/gks1219.

[32] Barrett, T., Clark, K., Gevorgyan, R., Gorelenkov, V., Gribov, E., Karsch-Mizrachi, I., Kimelman, M., Pruitt, K.D., Resenchuk, S., Tatusova, T., Yaschenko, E., and Ostell, J. (2012). "BioProject and BioSample databases at NCBI: facilitating capture and organization of metadata". *Nucleic Acids Research*, **40**(Database issue): D57–D63. doi:10.1093/nar/gkr1163.

[33] Coordinators, N.R. (2013). "Database resources of the National Center for Biotechnology Information". *Nucleic Acids Research*, **41**(Database issue): D8–D20. doi:10.1093/nar/gks1189.

[34] Altintas, I., Berkley, C., Jaeger, E., Jones, M., Ludascher, B., and Mock, S., "Kepler: an extensible system for design and execution of scientific workflows". In "Proceedings. 16th International Conference on Scientific and Statistical Database Management, 2004.", pp. 423–424 (IEEE, 2004). ISBN 0-7695-2146-0. doi:10.1109/SSDM.2004.1311241.

[35] Oinn, T., Addis, M., Ferris, J., Marvin, D., Senger, M., Greenwood, M., Carver, T., Glover, K., Pocock, M.R., Wipat, A., and Li, P. (2004). "Taverna: a tool for the composition and enactment of bioinformatics workflows". *Bioinformatics (Oxford, England)*, **20**(17): 3045–3054. doi:10.1093/bioinformatics/bth361.

[36] Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J.J., Nielsen, H.F., Karmarkar, A., and Lafon, Y. (2007). "SOAP Version 1.2 Part 1: Messaging Framework (Second Edition)". `http://www.w3.org/TR/soap12-part1/`.

[37] Fielding, R.T. and Taylor, R.N. (2002). "Principled design of the modern Web architecture". *ACM Transactions on Internet Technology*, **2**(2): 115–150. doi:10.1145/514183.514185.

[38] Stein, L.D. (2003). "Integrating biological databases." *Nature Reviews Genetics*, **4**(5): 337–345. doi:10.1038/nrg1065.

[39] Goble, C. and Stevens, R. (2008). "State of the nation in data integration for bioinformatics". *Journal of biomedical informatics*, **41**(5): 687–693. doi:10.1016/j.jbi.2008.01.008.

[40] Fowler, M. (2006). "Continuous integration". `http://martinfowler.com/articles/continuousIntegration.html`.

[41] Leinonen, R., Akhtar, R., Birney, E., Bower, L., Cerdeno-Tárraga, A., Cheng, Y., Cleland, I., Faruque, N., Goodgame, N., Gibson, R., Hoad, G., Jang, M., Pakseresht, N., Plaister, S., Radhakrishnan, R., Reddy, K., Sobhany, S., Ten Hoopen, P., Vaughan, R., Zalunin, V., and Cochrane, G. (2011). "The European Nucleotide Archive". *Nucleic Acids Research*, **39**(Database issue): D28–D31. doi:10.1093/nar/gkq967.

[42] Cochrane, G., Akhtar, R., Bonfield, J., Bower, L., Demiralp, F., Faruque, N., Gibson, R., Hoad, G., Hubbard, T., Hunter, C., Jang, M., Juhos, S., Leinonen, R., Leonard, S., Lin, Q., Lopez, R., Lorenc, D., McWilliam, H., Mukherjee, G., Plaister, S., Radhakrishnan, R., Robinson, S., Sobhany, S., Ten Hoopen, P., R., V., Zalunin, V., and Birney, E. (2008). "Petabyte-scale innovations at the european nucleotide archive". *Nucleic Acids Research*, **37**: 19–25.

[43] Hartmann, M., Howes, C.G., Veldre, V., Schneider, S., Vaishampayan, P.A., Yannarell, A.C., Quince, C., Johansson, P., Björkroth, K.J., Abarenkov, K., Hallam, S.J., Mohn, W.W., and Nilsson, R.H. (2011). "V-REVCOMP: automated high-throughput detection of reverse complementary 16S rRNA gene sequences in large environmental and taxonomic datasets". *FEMS Microbiology Letters*, **319**(2): 140–5.

[44] Eddy, S.R. (2011). "Accelerated profile hmm searches". *PLoS Computational Biology*, **7**(10): e1002195. doi:10.1371/journal.pcbi.1002195.

[45] Yilmaz, P., Kottmann, R., Field, D., Knight, R., Cole, J.R., Amaral-Zettler, L., Gilbert, J.a., Karsch-Mizrachi, I., Johnston, A., Cochrane, G., Vaughan, R., Hunter, C., Park, J., Morrison, N., Rocca-Serra, P., Sterk, P., Arumugam, M., Bailey, M., Baumgartner, L., Birren, B.W., Blaser, M.J., Bonazzi, V., Booth, T., Bork, P., Bushman, F.D., Buttigieg, P.L., Chain, Patrick S.G.and Charlson, E., Costello, E.K., Huot-Creasy, H., Dawyndt, P., DeSantis, T., Fierer, N., Fuhrman, J.A., Gallery, R.E., Gevers, D., Gibbs, R.A., San Gil, I., Gonzalez, A., Gordon, J.I., Guralnick, R., Hankeln, W., Highlander, S., Hugenholtz, P., Jansson, J., Kau, A.L., Kelley, S.T., Kennedy, J., Knights, D., Koren, O., Kuczynski, J., Kyrpides, N., Larsen, R., Lauber, C.L., Legg, T., Ley, R.E., Lozupone, C.A., Ludwig, W., Lyons, D., Maguire, E., Methé, B.A., Meyer, F., Muegge, B., Nakielny, S., Nelson, K.E., Nemergut, D., Neufeld, J.D., Newbold, L.K., Oliver, A.E., Pace, N.R., Palanisamy, G., Peplies, J., Petrosino,

J., Proctor, L., Pruesse, E., Quast, C., Raes, J., Ratnasingham, S., Ravel, J., Relman, D.A., Assunta-Sansone, S., Schloss, P.D., Schriml, L., Sinha, R., Smith, M.I., Sodergren, E., Spo, A., Stombaugh, J., Tiedje, J.M., Ward, D.V., Weinstock, G.M., Wendel, D., White, O., Whiteley, A., Wilke, A., Wortman, J.R., Yatsunenko, T., and Glöckner, F.O. (2011). "Minimum information about a marker gene sequence (MIMARKS) and minimum information about any (x) sequence (MIxS) specifications". *Nature Biotechnology*, **29**(5): 415–420. doi:10.1038/nbt.1823.

[46] Van Brabant, B., Gray, T., Verslyppe, B., Kyrpides, N., Dietrich, K., Glöckner, F.O., Cole, J., Farris, R., Schriml, L.M., De Vos, P., De Baets, B., Field, D., and Dawyndt, P. (2008). "Laying the foundation for a Genomic Rosetta Stone: creating information hubs through the use of consensus identifiers". *Omics: A Journal of Integrative Biology*, **12**(2): 123–127. doi: 10.1089/omi.2008.0020.

[47] Cole, J., Wang, Q., Cardenas, E., Fish, J., Chai, B., Farris, R., Kulam-Syed-Mohideen, A., McGarrell, D., Marsh, T., Garrity, G., and Tiedje, J. (2009). "The Ribosomal Database Project: improved alignments and new tools for rRNA analysis". *Nucleic Acids Research*, **37**(Database issue): D141–D145. doi:10.1093/nar/gkn879.

[48] Verslyppe, B., De Smet, W., De Baets, B., De Vos, P., and Dawyndt, P. (2013). "Straininfo introduces electronic passports for microorganisms". Manuscript submitted for publication.

[49] Kottmann, R., Kostadinov, I., Duhaime, M.B., Buttigieg, P.L., Yilmaz, P., Hankeln, W., Waldmann, J., and Glöckner, F.O. (2010). "Megx.net: integrated database resource for marine ecological genomics". *Nucleic Acids Research*, **38**(Database issue): D391–D395. doi:10.1093/nar/gkp918.

[50] Klyne, G. and Carroll, J.J. (2004). "Resource Description Framework (RDF): Concepts and Abstract Syntax". `http://www.w3.org/TR/rdf-concepts/`.

[51] Ludwig, W. and Klenk, H. (2001). "Overview: a phylogenetic backbone and taxonomic framework for prokaryotic systematics". *Bergey's Manual of Systematic Bacteriology*, **2**: 49–65.

[52] Janda, J.M. and Abbott, S.L. (2007). "16S rRNA gene sequencing for bacterial identification in the diagnostic laboratory: pluses, perils, and pitfalls". *Journal of Clinical Microbiology*, **45**(9): 2761–2764. doi:10.1128/JCM.01228-07.

[53] Yarza, P., Spröer, C., Swiderski, J., Mrotzek, N., Spring, S., Tindall, B.J., Gronow, S., Pukall, R., Klenk, H.P., Lang, E., Verbarg, S., Crouch, A., Lilburn, T., Beck, B., Unosson, C., Cardew, S., Moore, E.R., Gomila, M., Nakagawa, Y., Janssens, D., De Vos, P., Peiren, J., Suttels, T., Clermont, D., Bizet, C., Sakamoto, M., Iida, T., Kudo, T., Kosako, Y., Oshida, Y., Ohkuma, M., Arahal, D.R., Spieck, E., Pommerening Roeser, A., Figge, M., Park, D., Buchanan, P., Cifuentes, A., Munoz, R., Euzéby, J.P., Schleifer, K.H., Ludwig, W., Amann, R., Glöckner, F.O., and Rosselló-Móra, R. (2013). "Sequencing orphan species initiative (SOS): Filling the gaps in the 16S rRNA gene sequence database for all species with validly published names". *Systematic and Applied Microbiology*, **36**(1): 69–73. doi:10.1016/j.syapm.2012.12.006.

[54] De Vos, P. and Truper, H.G. (2000). "Judicial commission of the international committee on systematic bacteriology; ixth international (iums) congress of bacteriology and applied microbiology". *International Journal of Systematic and Evolutionary Microbiology*, **50**(6): 2239–2244. ISSN 1466-5026 1466-5034. doi:10.1099/00207713-50-6-2239.

[55] Fleischmann, R.D., Adams, M.D., White, O., Clayton, R.A., Kirkness, E.F., Kerlavage, A.R., Bult, C.J., Tomb, J.F., Dougherty, B.A., Merrick, J.M., McKenney, K., Sutton, G., FitzHugh, W., Fields, C., Gocayne, J.D., Scott, J., Shirley, R., Liu, L.I., Glodek, A., Kelley, J.M., Weidman, J.F., Phillips, C.A., Spriggs, T., Hedblom, E., Cotton, M.D., Utterback, T.R., Hanna, M.C., Nguyen, D.T., Saudek, D.M., Brandon, R.C., Fine, L.D., Fritchman, J.L., Fuhrmann, J.L., Geoghagen, N.S.M., Gnehm, C.L., McDonald, L.A., Small, K.V., Fraser, C.M., Smith, H.O., and Venter, J.C. (1995). "Whole-genome random sequencing and assembly of Haemophilus influenzae Rd". *Science*, **269**(5223): 496–512.

[56] Peterson, J., Garges, S., Giovanni, M., McInnes, P., Wang, L., Schloss, J.A., Bonazzi, V., McEwen, J.E., Wetterstrand, K.A., Deal, C., Baker, C.C., Di Francesco, V., Howcroft, T.K., Karp, R.W., Lunsford, R.D., Wellington, C.R.,

Belachew, T., Wright, M., Giblin, C., David, H., Mills, M., Salomon, R., Mullins, C., Akolkar, B., Begg, L., Davis, C., Grandison, L., Humble, M., Khalsa, J., Little, A.R., Peavy, H., Pontzer, C., Portnoy, M., Sayre, M.H., Starke-Reed, P., Zakhari, S., Read, J., Watson, B., and Guyer, M. (2009). "The NIH human microbiome project". *Genome Research*, **19**(12): 2317–23.

[57] Huson, D.H., Richter, D.C., Rausch, C., Dezulian, T., Franz, M., and Rupp, R. (2007). "Dendroscope: an interactive viewer for large phylogenetic trees". *BMC Bioinformatics*, **8**: 460.

[58] Coenye, T. and Vandamme, P. (2003). "Intragenomic heterogeneity between multiple 16S ribosomal RNA operons in sequenced bacterial genomes". *FEMS Microbiology Letters*, **228**(1): 45–49.

[59] Rastogi, R., Wu, M., DasGupta, I., and Fox, G.E. (2009). "Visualization of ribosomal RNA operon copy number distribution". *BMC Microbiology*, **9**(1): 208.

[60] Yarza, P., Ludwig, W., Euzéby, J., Amann, R., Schleifer, K.H., Glöckner, F.O., and Rosselló-Móra, R. (2010). "Update of the All-Species Living Tree Project based on 16S and 23S rRNA sequence analyses". *Systematic and Applied Microbiology*, **33**(6): 291–299. doi:10.1016/j.syapm.2010.08.001.

[61] Pei, A.Y., Oberdorf, W.E., Nossa, C.W., Agarwal, A., Chokshi, P., Gerz, E.a., Jin, Z., Lee, P., Yang, L., Poles, M., Brown, S.M., Sotero, S., Desantis, T., Brodie, E., Nelson, K., and Pei, Z. (2010). "Diversity of 16S rRNA genes within individual prokaryotic genomes". *Applied and Environmental Microbiology*, **76**(12): 3886–97.

[62] Cole, J.R., Chai, B., Farris, R.J., Wang, Q., Kulam-Syed-Mohideen, A.S., McGarrell, D.M., Bandela, A.M., Cardenas, E., Garrity, G.M., and Tiedje, J.M. (2006). "The Ribosomal Database Project (RDP-II): introducing myRDP space and quality controlled public data". *Nucleic Acids Research*, **35**(suppl 1): D169–D172.

[63] DeSantis, T.Z., Hugenholtz, P., Larsen, N., Rojas, M., Brodie, E.L., Keller, K., Huber, T., Dalevi, D., Hu, P., and Andersen, G.L. (2006). "Greengenes, a chimera-checked 16S rRNA gene database and workbench compatible

with ARB". *Applied and Environmental Microbiology*, **72**(7): 5069–5072. doi:10.1128/AEM.03006-05.

[64] Chun, J., Lee, J.H., Jung, Y., Kim, M., Kim, S., Kim, B.K., and Lim, Y.W. (2007). "EzTaxon: a web-based tool for the identification of prokaryotes based on 16S ribosomal RNA gene sequences". *International Journal of Systematic and Evolutionary Microbiology*, **57**: 2259–2261. doi:10.1099/ijs.0.64915-0.

[65] Lagesen, K., Hallin, P., Rødland, E.A., Stærfeldt, H.H., Rognes, T., and Ussery, D.W. (2007). "Rnammer: consistent and rapid annotation of ribosomal rna genes". *Nucleic Acids Research*, **35**(9): 3100–3108.

[66] Peplies, J., Kottmann, R., Ludwig, W., and Glöckner, F.O. (2008). "A standard operating procedure for phylogenetic inference (SOPPI) using (rRNA) marker genes". *Systematic and Applied Microbiology*, **31**(4): 251–257. doi:10.1016/j.syapm.2008.08.003.

[67] Turnbaugh, P.J., Ley, R.E., Hamady, M., Fraser-Liggett, C.M., Knight, R., and Gordon, J.I. (2007). "The Human Microbiome Project". *Nature*, **449**(7164): 804–810. doi:10.1038/nature06244.

[68] Gilbert, J.A., Bailey, M., Field, D., Fierer, N., Fuhrman, J.A., Hu, B., Jansson, J., Knight, R., Kowalchuk, G.A., Kyrpides, N.C., Meyer, F., and Stevens, R. (2011). "The Earth Microbiome Project: the meeting report for the 1st International Earth Microbiome Project Conference, Shenzhen, China, June 13th-15th 2011". *Standards in Genomic Sciences*, **5**(2): 243–247.

[69] Wooley, J.C., Godzik, A., and Friedberg, I. (2010). "A primer on metagenomics". *PLoS Computational Biology*, **6**(2): e1000667.

[70] Yarza, P., Richter, M., Peplies, J., Euzeby, J., Amann, R., Schleifer, K.H., Ludwig, W., Glöckner, F.O., and Rosselló-Móra, R. (2008). "The All-Species Living Tree Project: a 16S rRNA-based phylogenetic tree of all sequenced type strains". *Systematic and Applied Microbiology*, **31**(4): 241–250. doi:10.1016/j.syapm.2008.07.001.

[71] Curtis, T.P., Sloan, W.T., and Scannell, J.W. (2002). "Estimating prokaryotic diversity and its limits". *Proceedings of the National Academy of Sciences*, **99**(16): 10494–10499. doi:10.1073/pnas.142680199.

[72] Ash, C., Priest, F.G., and Collins, M.D. (1993). "Molecular identification of rRNA group 3 bacilli (Ash, Farrow, Wallbanks and Collins) using a PCR probe test". *Antonie van Leeuwenhoek*, **64**(3-4): 253–260.

[73] Ludwig, W., Strunk, O., Westram, R., Richter, L., Meier, H., Yadhukumar, Buchner, A., Lai, T., Steppi, S., Jobb, G., Förster, W., Brettske, I., Gerber, S., Ginhart, A.W., Gross, O., Grumann, S., Hermann, S., Jost, R., König, A., Liss, T., Lüssmann, R., May, M., Nonhoff, B., Reichel, B., Strehlow, R., Stamatakis, A., Stuckmann, N., Vilbig, A., Lenke, M., Ludwig, T., Bode, A., and Schleifer, K.H. (2004). "ARB: a software environment for sequence data". *Nucleic Acids Research*, **32**(4): 1363–71.

[74] Bouyssou, D., "Building criteria: a prerequisite for MCDA". In C.A. Bana e Costa, editor, "Readings in multiple criteria decision aid", pp. 58–80 (Springer-Verlag, 1990).

[75] Brosius, J., Palmer, M.L., Kennedy, P.J., and Noller, H.F. (1978). "Complete nucleotide sequence of a 16s ribosomal rna gene from escherichia coli". *Proceedings of the National Academy of Sciences*, **75**(10): 4801–4805.

[76] Stackebrandt, E. and Ebers, J. (2006). "Taxonomic parameters revisited: tarnished gold standards". *Microbiology Today*, **33**: 152–155.

[77] De Loof, K., De Baets, B., De Meyer, H., and Brüggemann, R. (2008). "A hitchhiker's guide to poset ranking". *Combinatorial Chemistry & High Throughput Screening*, **11**(9): 734–744.

[78] Davey, B. and Priestley, H., *Introduction to lattices and order* (Cambridge University Press, Cambridge, UK, 2002), 2nd edition.

[79] De Loof, K., De Meyer, H., and De Baets, B. (2006). "Exploiting the lattice of ideals representation of a poset". *Fundamenta Informaticae*, **71**(2): 309–321.

[80] De Loof, K., De Baets, B., and De Meyer, H. (2011). "Approximation of average ranks in posets". *MATCH*, **66**: 219–229.

[81] Field, D., Garrity, G., Gray, T., Morrison, N., Selengut, J., Sterk, P., Tatusova, T., Thomson, N., Allen, M.J., Angiuoli, S.V., Ashburner, M., Axelrod, N., Baldauf, S., Ballard, S., Boore, J., Cochrane, G., Cole, J., Dawyndt, P., De

Vos, P., DePamphilis, C., Edwards, R., Faruque, N., Feldman, R., Gilbert, J., Gilna, P., Glöckner, F.O., Goldstein, P., Guralnick, R., Haft, D., Hancock, D., Hermjakob, H., Hertz-Fowler, C., Hugenholtz, P., Joint, I., Kagan, L., Kane, M., Kennedy, J., Kowalchuk, G., Kottmann, R., Kolker, E., Kravitz, S., Kyrpides, N., Leebens-Mack, J., Lewis, S.E., Li, K., Lister, A.L., Lord, P., Maltsev, N., Markowitz, V., Martiny, J., Methe, B., Mizrachi, I., Moxon, R., Nelson, K., Parkhill, J., Proctor, L., White, O., Sansone, S.A., Spiers, A., Stevens, R., Swift, P., Taylor, C., Tateno, Y., Tett, A., Turner, S., Ussery, D., Vaughan, B., Ward, N., Whetzel, T., San Gil, I., Wilson, G., and Wipat, A. (2008). "The minimum information about a genome sequence (MIGS) specification". *Nature Biotechnology*, **26**(5): 541–547. doi:10.1038/nbt1360.

[82] Matias, F. and Rodrigues, M.F.d.A. (2011). "New PHA products using unrelated carbon sources". *Brazilian Journal of Microbiology*, **42**: 1354–1363.

[83] McDonald, D., Price, M.N., Goodrich, J., Nawrocki, E.P., Desantis, T.Z., Probst, A., Andersen, G.L., Knight, R., and Hugenholtz, P. (2012). "An improved Greengenes taxonomy with explicit ranks for ecological and evolutionary analyses of bacteria and archaea". *The ISME Journal*, **6**(3): 610–8.

[84] Ashelford, K.E., Chuzhanova, N.A., Fry, J.C., Jones, A.J., and Weightman, A.J. (2005). "At least 1 in 20 16S rRNA sequence records currently held in public repositories is estimated to contain substantial anomalies". *Applied and Environmental Microbiology*, **71**(12): 7724–7736. doi:10.1128/AEM.71.12.7724-7736.2005.

[85] Huber, T., Faulkner, G., and Hugenholtz, P. (2004). "Bellerophon: a program to detect chimeric sequences in multiple sequence alignments". *Bioinformatics*, **20**(14): 2317–2319.

[86] Yoon, J.H., Kang, S.J., Oh, H.W., and Oh, T.K. (2006). "Stenotrophomonas dokdonensis sp. nov., isolated from soil". *International Journal of Systematic and Evolutionary Microbiology*, **56**: 1363–1367. doi:10.1099/ijs.0.64091-0.

[87] Lee, D.S., Ryu, S.H., Hwang, H.W., Kim, Y.J., Park, M., Lee, J.R., Lee, S.S., and Jeon, C.O. (2008). "Pseudoxanthomonas sacheonensis sp. nov.,

isolated from BTEX-contaminated soil in Korea, transfer of Stenotrophomonas dokdonensis Yoon et al. 2006 to the genus Pseudoxanthomonas as Pseudoxanthomonas dokdonensis comb. nov. and emended description of the genus Pseudoxanthomonas". *International Journal of Systematic and Evolutionary Microbiology*, **58**: 2235–2240. doi:10.1099/ijs.0.65678-0.

[88] Lipman, D.J. and Pearson, W.R. (1985). "Rapid and sensitive protein similarity searches". *Science*, **227**(4693): 1435–1441. doi:10.1126/science.2983426.

[89] Pearson, W.R. and Lipman, D.J. (1988). "Improved tools for biological sequence comparison". *Proceedings of the National Academy of Sciences*, **85**(8): 2444–2448. doi:10.1073/pnas.85.8.2444.

[90] IUPAC, *Compendium of Chemical Terminology, 2nd ed. (the "Gold Book")* (Blackwell Scientific Publications (Oxford), 1997). ISBN 0-9678550-9-8. doi:10.1351/goldbook.N04254. XML on-line corrected version: `http://goldbook.iupac.org` (2006-) created by M. Nic, J. Jirat, B. Kosata; updates compiled by A. Jenkins.

[91] Altschul, S.F., Gish, W., Miller, W., Myers, E.W., and Lipman, D.J. (1990). "Basic local alignment search tool". *Journal of Molecular Biology*, **215**(3): 403–410. doi:10.1016/S0022-2836(05)80360-2.

[92] Huse, S.M., Huber, J.A., Morrison, H.G., Sogin, M.L., and Welch, D.M. (2007). "Accuracy and quality of massively parallel DNA pyrosequencing." *Genome Biology*, **8**(7): R143. doi:10.1186/gb-2007-8-7-r143.

[93] Baldauf, S.L. (2003). "Phylogeny for the faint of heart: a tutorial". *Trends in Genetics*, **19**(6): 345–351. doi:10.1016/S0168-9525(03)00112-4.

[94] Letunic, I. and Bork, P. (2007). "Interactive Tree Of Life (iTOL): an online tool for phylogenetic tree display and annotation". *Bioinformatics (Oxford, England)*, **23**(1): 127–128. doi:10.1093/bioinformatics/btl529.

[95] Letunic, I. and Bork, P. (2011). "Interactive Tree Of Life v2: online annotation and display of phylogenetic trees made easy". *Nucleic Acids Research*, **39**(Web Server issue): W475–478. doi:10.1093/nar/gkr201.

[96] Whitman, W., Goodfellow, M., Kämpfer, P., Busse, H., Trujillo, M., Ludwig, W., Suzuki, K., and Parte, A., *Bergey's Manual of Systematic Bacteriology*. Bergey's Manual of Systematic Bacteriology (Springer New York, 2012). ISBN 9780387682334.

[97] Garrity, G., Lilburn, T., Cole, J., Harrison, S., Euzéby, J., and Tindall, B. "Taxonomic outline of the bacteria and archaea, release 7.7". doi: 10.1601/TOBA7.7.

[98] Jones, N.C. and Pevzner, P.A., *An introduction to bioinformatics algorithms*. computational molecular biology (The MIT Press, 2004).

[99] Sneath, P.H., Sokal, R.R., et al., *Numerical taxonomy. The principles and practice of numerical classification* (W.H. Freeman and Company (San Francisco), 1973).

[100] Saitou, N. and Nei, M. (1987). "The neighbor-joining method: a new method for reconstructing phylogenetic trees". *Molecular Biology and Evolution*, **4**(4): 406–425.

[101] Fitch, W. (1971). "Toward defining the course of Evolution: Minimum change for a specific tree topology". *Systematic Biology*, **20**(4): 406–416. doi:10.1093/sysbio/20.4.406.

[102] Felsenstein, J. (1981). "Evolutionary trees from DNA sequences: A maximum likelihood approach". *Journal of Molecular Evolution*, **17**(6): 368–376. doi:10.1007/BF01734359.

[103] Hillis, D., Huelsenbeck, J., and Cunningham, C. (1994). "Application and accuracy of molecular phylogenies". *Science*, **264**(5159): 671–677. doi:10.1126/science.8171318.

[104] Sievers, F., Wilm, A., Dineen, D., Gibson, T.J., Karplus, K., Li, W., Lopez, R., McWilliam, H., Remmert, M., Söding, J., Thompson, J.D., and Higgins, D.G. (2011). "Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega." *Molecular Systems Biology*, **7**(539). doi:10.1038/msb.2011.75.

[105] Fowler, M., *Patterns of enterprise application architecture* (Pearson Education Inc., 2003).

[106] Gamma, E., Helm, R., Johnson, R., and Vlissides, J., *Design Patterns: elements of reusable object-oriented software* (Addison-Wesley, 1995).

[107] Goetz, B., Peirls, T., Bloch, J., Bowbeer, J., Holmes, D., and Lea, D., *Java concurrency in practice* (Addison-Wesley, 2006).

[108] Turenne, C.Y., Tschetter, L., Wolfe, J., and Kabani, A. (2001). "Necessity of quality-controlled 16S rRNA gene sequence databases: Identifying non-tuberculous mycobacterium species". *Journal of Clinical Microbiology*, **39**(10): 3637–3648. doi:10.1128/JCM.39.10.3638-3648.2001.

[109] Kottmann, R., Gray, T., Murphy, S., Kagan, L., Kravitz, S., Lombardot, T., Field, D., and Glöckner, F.O. (2008). "A standard MIGS/MIMS compliant XML Schema: toward the development of the Genomic Contextual Data Markup Language (GCDML)". *Omics: a journal of integrative biology*, **12**(2): 115–121. doi:10.1089/omi.2008.0A10.

List of publications

- Verslyppe, B., Slabbinck, B., **De Smet, W.**, De Vos, P., De Baets, B., and Dawyndt, P., "Staininfo.net web services: Enabling microbiologic workflows such as phylogenetic tree building and biomarker comparison". In *eScience, 2008. eScience '08. IEEE Fourth International Conference on*, pp. 603–607 (2008).

- Verslyppe, B., Kottmann, R., **De Smet, W.**, De Baets, B., De Vos, P., and Dawyndt, P. (2010). "Microbiological Common Language (MCL): a standard for electronic information exchange in the microbial commons". *Research in Microbiology*, **161**(6): 439–445.

- Verslyppe, B., **De Smet, W.**, De Vos, P., De Baets, B., and Dawyndt, P., "Semantic integration of isolation habitat and location in staininfo". In *BMC Bioinformatics*, volume 11, p. 2 (2010).

- Verslyppe, B., **De Smet, W.**, De Baets, B., De Vos, P., and Dawyndt, P. (2011). "Make Histri: reconstructing the exchange history of bacterial and archaeal type strains". *Systematic and Applied Microbiology*, **34**(5): 328–36.

- **De Smet, W.**, De Loof, K., De Vos, P., Dawyndt, P., and De Baets, B. (2013). "Filtering and ranking techniques for automated selection of high-quality 16s rRNA gene sequences". *Systematic and Applied Microbiology*. Article accepted for publication, Sept 10, 2013.

- **De Smet, W.**, De Baets, B., De Loof, K., De Vos, P., and Dawyndt, P. (2013). "Seqrank: quality assessment and ranking of 16s rrna gene sequences in staininfo". Application note submitted for publication.

- **De Smet, W.**, Kottmann, R., Cole, J., Field, D., Huot Creasy, H., Kyrpides, N.C., Schriml, L., Tatusova, T., Ussery, D., White, O., Glöckner, F.O.,

De Vos, P., De Baets, B., and Dawyndt, P. (2013). "The genomic rosetta stone: resolving curated metadata through stable identifiers". Article in preparation.

- Verslyppe, B., **De Smet, W.**, De Baets, B., De Vos, P., and Dawyndt, P. (2013). "Straininfo introduces electronic passports for microorganisms". Article submitted for publication.